KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Technology

The automatic blind spot camera: hard real-time detection of moving objects from a moving camera

Kristof VAN BEECK

Supervisors:
Prof. dr. ir. Toon Goedemé
Prof. dr. ir. Tinne Tuytelaars

Dissertation presented in partial fulfilment
of the requirements for the degree
of Doctor in Engineering Technology

September 2016

The automatic blind spot camera: hard real-time detection of moving objects from a moving camera

Kristof VAN BEECK

Examination committee:
Prof. dr. Guido Aerts, chair
Prof. dr. ir. Toon Goedemé, supervisor
Prof. dr. ir. Tinne Tuytelaars, co-supervisor
Prof. dr. Joost Vennekens
Prof. dr. ir. Luc Van Eycken
Prof. dr. Stijn Daniels
Prof. dr. ir. Bart Vanrumste
Dr. ir. Rob Wijnhoven
Prof. dr. ir. Antonio M. López
(Universitat Autònoma de Barcelona (UAB))

Dissertation presented in partial fulfilment of the requirements for the degree of Doctor in Engineering Technology

September 2016

Preface

Looking back over the past six years, I could never imagine that this would be such an amazing journey. Initially I was told that, apart from the academic degree, obtaining a PhD for a large part entails some form of personal enrichment. I now thoroughly understand that statement.

Throughout these years, I had the opportunity to meet many interesting people in the professional context of several computer vision conferences. These countless discussions, where our research results were published and discussed, often gave me important insights and new ideas which enabled me to explore different paths regarding my own research. Apart from these professional meetings, this dissertation offered me the privilege to meet different cultures and visit places all over the world which I otherwise would have never seen. I clearly remember my first conference in 2011, which was held in Japan only a few months after the earthquake, tsunami and the following Fukushima nuclear disaster. Needless to say that this was a tremendous experience. This conference paved the way for numerous other life-lasting memories: seeing the Great Wall of China, having dinner with colleagues at the Timberline Lodge at Mount Hood in Portland, Oregon, visiting the famous Petronas Towers in Kuala Lumpur and so on. None of this – both personally and professionally – would have been possible without the help and support of numerous people along the way, which I would like to thank sincerely.

First of all I would like to thank my supervisor, Prof. Toon Goedemé who made all of this possible. Toon, thank you for giving me the opportunity to undertake this challenge and for your continued support throughout these years. I still remember our meeting several years ago in our office couch thinking about a good name for the new research group which you just started. Since then, the EAVISE research team kept on expanding year after year. Your comments and feedback on my research work were invaluable, and each paper review meant a significant improvement in quality. You made it possible for me to finish my PhD without ever having to worry about administrative work – such as finding

research funds – for which I am very grateful.

I would like to thank my co-supervisor Prof. Tinne Tuytelaars who was of great help, and who during the first years of my PhD acted as supervisor. Her key insights and critical comments during each of our meetings were of great importance to my research. Furthermore I am thankful to each member of my examination committee for the time they invested in proofreading this dissertation, and their valuable feedback. Their efforts undeniably made this work more complete.

Of course, a positive work environment is of crucial importance. Therefore I am also very thankful to all of my colleagues for the nice atmosphere, the entertaining discussions during the coffee breaks and all non-work related activities which I thoroughly enjoyed. For their specific contribution to this dissertation I would like to especially thank the following colleagues. Floris, who co-authored two important publications. Dries, who acted as regular truck driver for several of my experiments, even during weekends and holidays. And finally Stijn and Steven who, as direct office mates, were the first to help me out with several technical and practical issues and were involved in many interesting discussions.

Natuurlijk wil ik ook met heel mijn hart mijn familie en schoonfamilie bedanken voor alle steun die ze mij geven, en voor de vele leuke familiemomenten. In het bijzonder wil ik graag mijn ouders bedanken die me de kans hebben geven om deze studies aan te vangen, en die mij mee gevormd hebben tot de persoon die ik nu ben. Helaas kan mijn vader dit alles niet meer bewust meemaken, maar ik ben er zeker van dat hij enorm trots op mij zou zijn. Uiteraard wil ik ook graag mijn broer en zijn gezin bedanken, van wie ik het voorrecht kreeg om trotse peter te worden van hun eerste zoontje Niels.

Mijn dank gaat ook uit naar al mijn vrienden bij wie ik steeds terecht kan voor een gezellig avondje uit of om mijn gedachten te verzetten. Iedereen opsommen zonder iemand te vergeten zou bijna onmogelijk zijn. Bedankt iedereen!

Tot slot wil ik graag de belangrijkste persoon in mijn leven bedanken. Marlies, bedankt om er steeds voor mij te zijn! Jouw onvoorwaardelijke steun heeft er mee voor gezorgd dat ik dit bereikt heb.

Abstract

Each year traffic accidents caused by the blind spot zone of trucks are responsible for an estimate of about 1300 casualties in Europe alone. These accidents almost always occur in a similar fashion: the truck driver takes a right hand turn at an intersection and overlooks vulnerable road users (VRUs – e.g. pedestrians or bicyclists) which continue their way straight ahead. They often have one of two distinctive causes: inattentiveness of the truck driver and/or the fact that these victims were located in a blind spot zone around the truck. To cope with these blind spot zones, several commercial systems were developed. However, each of them has specific disadvantages, and as such none of them seems to handle the blind spot problem completely. The most widely used safety systems are the – since 2003 obliged by law – blind spot mirrors. Furthermore, often blind spot cameras are employed. These systems display the blind spot zone on a monitor in the truck's cabin – using wide-angle lenses – when a right hand turn is signalled. More recently, active safety systems are used (such as ultrasonic distance sensors), which automatically generate an alarm towards the truck driver. However, these systems fail to distinguish static objects (i.e. traffic signs) from VRUs and often generate false positive alarms.

The aforementioned commercially available systems fail to reduce the number of casualties. Therefore, in this dissertation we describe an active safety system relying solely on the input images from the blind spot camera. Using computer vision object detection methodologies, our system is able to efficiently detect VRUs in the challenging blind spot images, and automatically warns the truck driver of their presence. Such a system has several advantages: it is always adjusted correctly, is easily integratable in existing passive blind spot camera setups, does not rely on the attentiveness of the truck driver and is able to distinguish VRUs from static objects. However, developing such a safety system is not an easy task. These VRUs are *multiclass* (they consist of pedestrians, bicyclists, children and so on) and appear in very diverse viewpoints and poses. Additionally, we need to cope with the large viewpoint and lens distortion induced by traditional blind spot cameras. Finally, our specific application

inherently requires extremely stringent demands with respect to the detection accuracy, throughput and latency. Indeed, excellent accuracy results need to be achieved for such a system to be usable in real-life scenarios, at real-time processing speeds. However, assuring hard real-time detection behaviour contradicts with the requirement for high detection accuracy. Specifically, object detection methodologies often require significant computational power to achieve high accuracy. As such, traditionally a trade-off exists between accuracy and throughput when only limited hardware is available. This is unfeasible for our application: our active safety system should achieve excellent accuracy results and at the same time should run in real-time on low-cost embedded hardware.

In this PhD we developed a methodology that eliminates this trade-off. The advantage of this contribution is two-fold. First, it allows for the detection of VRUs in challenging images where existing object detectors fail (due to the specific viewpoint and lens distortion). Second, this approach enables the use of highly accurate object detection methodologies which would otherwise be too time consuming. As such, we achieve excellent accuracy at real-time processing speeds. To validate this, we acquired a unique and valuable dataset recorded with a genuine blind spot camera mounted on a real truck in which several dangerous blind spot scenarios were simulated. This dataset increased in size and complexity throughout this dissertation.

This initial methodology enabled the efficient detection and tracking of pedestrians in our blind spot camera images. We proved that this methodology easily generalises itself to other scenarios with similar viewpoint. Furthermore, we presented additional contributions towards an increase in detection accuracy. We developed a methodology that enables the efficient combination of multiple pedestrian detectors, extended our initial approach to better model the specific distortion and developed a method to enable multiclass detection. Finally, we further optimised and integrated the aforementioned methodologies, and presented a final vision-based only active safety system for the blind spot zone. We conclude that our final active safety system manages to meet the stringent accuracy and latency demands required for such a system to be usable in practice.

Beknopte samenvatting

Elk jaar vallen er in Europa alleen al ongeveer 1300 slachtoffers te betreuren omwille van de dode hoek bij vrachtwagens. Deze ongevallen gebeuren bijna altijd op dezelfde manier: de vrachtwagenbestuurder slaat rechtsaf aan een kruispunt of rotonde en ziet zwakke weggebruikers over het hoofd die hun weg rechtdoor vervolgen. Vaak liggen er twee specifieke oorzaken aan de basis van deze ongevallen: onoplettendheid van de vrachtwagenbestuurder en/of het feit dat de zwakke weggebruikers zich in de dode hoek van de vrachtwagen bevinden. Er werden verschillende commerciële systemen ontwikkeld om dit probleem aan te pakken. Alle bestaande systemen hebben echter verschillende specifieke nadelen, waardoor er momenteel geen systeem bestaat dat de dodehoekproblematiek volledig kan oplossen. De meest gebruikte oplossing blijft een – sinds 2003 bij wet verplichte – dodehoekspiegel. Verder wordt er ook vaak gebruik gemaakt van een dodehoekcamera. Deze toont via een breedhoeklens de dodehoekzone op een beeldscherm in de vrachtwagencabine wanneer een afslag naar rechts wordt aangegeven. Recent wordt er meer en meer gebruikt gemaakt van actieve systemen, zoals bijvoorbeeld ultrasone afstandssensoren. Deze systemen geven automatisch een alarm aan de vrachtwagenbestuurder. Het grote nadeel van dergelijke systemen is dat ze geen onderscheid kunnen maken tussen statische objecten (zoals bijvoorbeeld een verkeersbord) en zwakke weggebruikers. Hierdoor worden er vaak valse alarmen gegenereerd.

De bovenstaande commerciële systemen blijken niet in staat om het aantal slachtoffers te reduceren. Daarom werd in dit doctoraat een actief veiligheidssysteem ontwikkeld dat enkel gebruik maakt van de beelden van de dodehoekcamera. Door gebruik te maken van computervisie-methodologieën (meer bepaald objectdetectietechnieken) is ons systeem in staat om op een efficiente manier zwakke weggebuikers te detecteren in deze uitdagende beelden en om automatisch de vrachtwagenbestuurder op de hoogte te brengen van hun aanwezigheid. Dit systeem heeft verschillende voordelen: het is steeds correct afgesteld, vertrouwt niet op de oplettendheid van de vrachtwagenbestuurder en kan impliciet een onderscheid maken tussen zwakke weggebruikers en statische

objecten. Het is echter niet eenvoudig om zo'n systeem te ontwikkelen. De categorie zwakke weggebruikers bestaat uit verschillende entiteiten (voetgangers, fietsers, kinderen enzovoort) die voorkomen in heel diverse kijkhoeken en houdingen. Bovendien moet het systeem kunnen omgaan met de grote kijkhoek en lensvervorming die wordt geïntroduceerd door deze traditionele dodehoekcamera's. Tot slot stelt onze specifieke applicatie inherent zeer strikte eisen wat betreft de detectie-accuraatheid, verwerkingssnelheid en reactietijd. Een uitstekende accuraatheid moet behaald worden om zo'n systeem in reële situaties te kunnen gebruiken en dit tegen real-time verwerkingstijd. Het verzekeren van hard real-time gedrag contradicteert echter met de eis voor hoge accuraatheid. Meer specifiek eisen objectdetectie-methodologieën vaak significante rekenkracht om een hoge accuraatheid te behalen. Hierdoor ontstaat de traditionele trade-off tussen accuraatheid en snelheid wanneer op gelimiteerde hardware gewerkt wordt. Dit is ontoelaatbaar voor onze applicatie: zo'n actief veiligheidssysteem moet een uitstekende accuraatheid behalen tegen real-time verwerkingssnelheden op low-cost hardware.

In dit doctoraat werd een methodologie ontwikkeld die de bovenstaande trade-off elimineert. Het voordeel van deze contributie is tweeledig. Ten eerste laat deze methodologie de detectie toe van zwakke weggebruikers in uitdagende afbeeldingen waar bestaande object detectoren falen (door de specifieke kijkrichting en lensvervorming). Ten tweede laat deze benadering het gebruik van zeer accurate objectdetectietechnieken toe, die anders te tijdsintensief zouden zijn. Hierdoor behalen we een uitstekende accuraatheid tegen real-time verwerkingssnelheden. Om onze methodologieën te valideren hebben we een unieke en waardevolle dataset opgenomen met een commerciële dodehoekcamera gemonteerd op een echte vrachtwagen. Hiervoor werden verschillende vaak voorkomende gevaarlijke dodehoeksituaties gesimuleerd. Deze dataset nam doorheen deze dissertatie toe in zowel grootte als complexiteit.

Onze intiële methodologie liet het efficient detecteren en volgen van voetgangers toe in onze dodehoekcamera-beelden. We hebben bewezen dat deze methodologie zichzelf gemakkelijk generaliseert naar andere scenario's met een soortgelijke kijkrichting. Verder hebben we additionele contributies gepresenteerd om de detectie-accuraatheid te verhogen. We ontwikkelden een methodologie die toelaat om op een efficiente manier meerdere persoonsdetectoren te combineren, we hebben onze intiële benadering uitgebreid om de specifieke vervorming beter te modelleren en we ontwikkelden een manier die *multiclass* detectie toelaat. Ten slotte hebben we bovenstaande methodologieën geoptimaliseerd en geïntegreerd om zo tot een finaal visie-gebaseerd actief veiligheidssysteem voor de dodehoekzone te komen. We concluderen dat ons finaal veiligheidssysteem erin slaagt om te voldoen aan de strenge eisen wat betreft de accuraatheid en reactietijd dat zo'n systeem moet behalen om in de praktijk bruikbaar te zijn.

Glossary

ACF Aggregated Channel Features. Pedestrian detection methodology [30], based on ICF.

ACPR Asian Conference on Pattern Recognition.

AP Average precision. The precision averaged over all values of the recall (i.e. identical to the AUC).

ATINER Athens Institute for Education and Research.

AUC Area Under the Curve. Often indicates the area under a PR curve, and thus a measure of accuracy.

BIVV Belgian Road Safety Institute (*Belgisch Instituut Voor de Verkeersveiligheid*).

CAVIAR Context Aware Vision using Image-based Active Recognition project.

CNN Convolutional Neural Networks. See R-CNN.

CPU Central Processing Unit.

CUDA Compute Unified Device Architecture, a GPGPU technology enabling the executing of algorithms on GPU.

CVICGTA Computer Vision, Imaging and Computer Graphics: Theory and Applications.

CVPR IEEE Conference on Computer Vision and Pattern Recognition.

CVRSUAD Computer Vision for Road Scene Understanding and Autonomous Driving. A workshop in conjunction with ECCV.

DPM Deformable Part Models. An object detection methodology which employs a non-rigid template model [44].

ECCV European Conference on Computer Vision.

EVW Embedded Vision Workshop. A workshop in conjunction with CVPR.

FN False Negatives. Evaluation metric of an object detection algorithm, indicating instances that are incorrectly classified as not belonging to the class to be detected.

FP False Positives. Evaluation metric of an object detection algorithm, indicating instances that are incorrectly classified as belonging to the class to be detected.

FPDW Fastest Pedestrian Detector in the West. Pedestrian detection methodology based on ICF [31].

FPPI False Positives Per Image. The average number of FP per evaluation image.

FPR False Positive Rate. Evaluation metric of an object detection algorithm. An indication of the number of false alarms.

FPS Frames per second. Indicates the throughput (i.e. speed) of a camera or an object detection algorithm in the context of this dissertation.

GCM Gross Combination Mass. The maximum allowable combined mass of a truck.

GMM Gaussian Mixture Models. A background subtraction methodology.

GPGPU General-Purpose computing on Graphics Processing Units, i.e. executing computer programs on the GPU to exploit parallelisation.

GPS Global Positioning System.

GPU Graphics Processing Units.

HGV Heavy Good Vehicle. A term indicating any truck with a GCM of 3500 kilograms.

HOG Histograms of Oriented Gradients. A pedestrian detection methodology presented in [21].

IAPR International Association of Pattern Recognition.

ICF Integral Channel Features. A pedestrian detection methodology presented in [32].

ICINCO International Conference on Informatics in Control, Automation and Robotics.

ICPR International Conference on Pattern Recognition.

ICPRAM International Conference on Pattern Recognition Applications and Methods.

INRIA Institut National de Recherche en Informatique et en Automatique.

LatSVM Latent SVM. Pedestrian detection methodology. Alternative name for DPM.

LatSVM-CC The cascaded version of LatSVM.

LNCS Lecture Notes in Computer Science.

LNEE Lecture Notes in Electrical Engineering.

LUF Lookup Function. A 2D function which contains calibration data, such as the height or width of a specific object (e.g. pedestrians) at a specific position.

LWIR Long Wave Infrared.

MPEG2 Motion Picture Experts Group 2. A compression standard.

MR Miss rate. Evaluation metric of an object detection algorithm. An indication of the percentage of all annotations that are not retrieved.

MVA IAPR Conference on Machine Vision Applications.

NMS Non-maxima Suppression. A technique to reduce the number of bounding boxes after a sliding window evaluation. In the case of overlapping bounding boxes (based on an overlap criterion) only the highest scoring bounding box is maintained.

PR curve Precision-recall curve. A curve which visualises the trade-off between the precision (fraction of the retrieved instances that are relevant) and the recall (fraction of relevant instances that are retrieved).

R-CNN Regional Convolution Neural Networks. An object detection methodology which uses region proposals as input to a deep learning architecture [49].

ROI Region Of Interest. A specific region in the image which might be of interest (e.g. containing a pedestrian).

SVM Support Vector Machines. A classification methodology.

SWOV Stichting Wetenschappelijk Onderzoek Verkeersveiligheid.

TN True Negatives. Evaluation metric of an object detection algorithm, indicating instances that are correctly classified as not belonging to the class to be detected.

TP True Positives. Evaluation metric of an object detection algorithm, indicating instances that are correctly classified as belonging to the class to be detected.

TRL Transport Research Laboratory.

TWMV Two wheeled motor vehicles.

UB Upper Body.

VGA Video Graphics Array, indicating a resolution of 640×480.

VGG Visual Geometry Group. Often refers to the deep convolutional neural networks developed by this research group.

VISAPP International Conference on Computer Vision Theory and Applications.

VJ Viola & Jones. An object detection methodology [117].

VRU Vulnerable Road Users. A term indicating road users which have a higher risk of being injured or killed in a traffic accident. Often considered to be pedestrians, bicyclists and mopeds.

WSPD Warp Speed Pedestrian Detector. Our hybrid CPU/GPU implementation presented in chapter 4, section 4.3.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to our ever increasing technological society, the global number of cameras drastically increases each year. The increasing number of surveillance cameras – currently about one surveillance camera is installed for every 35 people worldwide – is only partially responsible for this. Millions of images are captured every day using smartphones and tablets. A significant amount of this data is made available online, through means of social media or public video websites. This vast amount of data opens up a wide variety of possibilities for the *computer vision* community.

Object detection, a subdomain of computer vision, aims to detect specific instances of one or more objects in images. Take for example the image shown in figure 1.1. For humans it is evident to recognise a work environment with a cluttered desk, two computer monitors, a coffee mug and so on. Our human brain is able to process and analyse the incoming visual information from our eyes in an extremely efficient way. However, for a computer this image is solely composed of millions of *pixels* indicating specific colour values. Identifying objects in these simple numbers remains a very challenging task. Nonetheless, recent state-of-the-art object detection algorithms perform reasonably well even on these complex images. The qualitative output of such an algorithm is visualised on the same image, using the colour coded bounding boxes (we employed R-CNN for this example [49], see chapter 3 for more detailed technical information). Reasonable recognition results are obtained. However, several objects are not detected and the actual analysis of this image took around 10 minutes (single core CPU only). Object detection algorithms in general work as follows. Based on a *training set* consisting of a number of *positive* images (containing the object of interest) and *negative* images (e.g. background

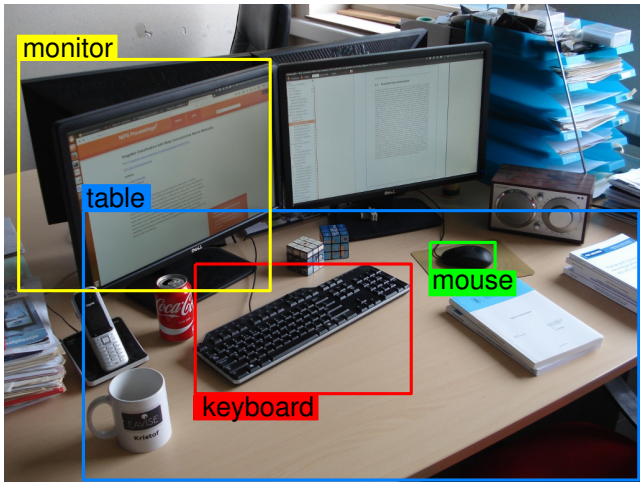


Figure 1.1: Resulting output from a state-of-the-art object detection algorithm (R-CNN). The four highest scoring objects are shown. Reasonable accuracy results are obtained.

images) a representative model of the object is learned using a *machine learning* method. Keep in mind that the detection of e.g. pedestrians in images is a specialised case of object detection in general. In such context, pedestrians are also viewed as *objects*. This learned model needs to generalise to other instances of the same object not present in the training set, while still being specific enough to only detect the objects of interest. This is difficult since the appearance of most objects varies significantly depending on the viewpoint and lighting conditions. Furthermore, specific objects introduce additional challenges. For example, a high variability in the appearance of pedestrians exists due to e.g. different clothes and poses. Over the past decade the accuracy of object detection algorithms on renowned, publicly available datasets increased significantly. The output detections on the example image shown above were found using a *deep learning* architecture, currently a state-of-the-art approach. This object detection methodology is able to achieve impressive accuracy results. Such algorithms are often trained on very large datasets (for example [94]). However, these deep learning methods rely on heavy hardware: multicore CPU and extensive GPU hardware is needed to perform all calculations in reasonable time limits. Even then, training such a network takes multiple days.

In this dissertation we primarily focus on detecting specific *objects*, namely vulnerable road users (further mentioned as VRUs). These mainly consist of pedestrians and bicyclists. Pedestrian detection is a widely researched

topic within the computer vision community due to the multitude of possible application domains. Apart from the traffic safety aspect, pedestrian detection enables for example human-robot interaction [119], human-human interaction analysis [23], abnormal behaviour detection [128], surveillance applications [83], people counting [97], autonomous lecture recordings [61], health applications [29] and so on. Concerning accuracy, a similar trend is noticeable in the pedestrian detection domain, where the detection accuracy keeps on increasing each year [7]. However, recent work indicates that a significant performance difference still exists between current pedestrian detection algorithms and human performance on the same images [126].

One could argue that the precision of currently available pedestrian detection algorithms is high enough for numerous applications, and that they are already usable for several practical problems. This is only partially true, e.g. when image analysis can happen offline and/or the detection accuracy is not critical. Examples of this are the tracking of people in sport games for subsequent game analysis or the safeguarding of privacy by offline automatic blurring of faces in mobile mapping images [88]. For safety critical applications however, a high accuracy and fast response time is essential. These stringent demands often make it unfeasible to employ out-of-the-box pedestrian detection algorithms. Fast detection speeds can be achieved when using specialised hardware, such as CPU clusters. However, this is not always possible. In practical applications, often only embedded low-cost and lightweight hardware platforms can be used (e.g. when performing object detection on drones [25]).

This dissertation focuses on a very specific safety critical application: using computer vision techniques that enable the detection of vulnerable road users in the blind spot zone of trucks. In essence, we aim to develop an automatic alarm system, which warns the truck driver when VRUs are present in the blind spot zone. As input, this alarm system solely relies on a monocular commercial blind spot camera. As such, this system should be easily integratable in existing passive blind spot camera setups. Evidently, real-time operation with very high accuracy is required. Furthermore, the algorithms that we developed ought to run in real-time on limited hardware. This is a challenging task, due to multiple reasons as will be discussed in chapter 2. An evident difference with existing techniques is found in the viewpoint. Existing pedestrian detection benchmark datasets often rely on a trivial forward-looking viewpoint, which significantly simplifies the problem. Figure 1.2 shows the difference between a widely used dataset (Caltech [34]) and the images that we captured in our own experiments. Significant viewpoint and lens distortion artifacts are observed. Existing commercial systems (such as ultrasonic sensors) seem not able to cope with this problem completely, as discussed in chapter 2 and chapter 3.

In the remainder of this chapter we give a summary of the main contributions

(a) Publicly available dataset (b) Our dataset

Figure 1.2: The difference in viewpoint between the publicly available datasets and our application imposes a significant additional challenge. Left: traditionally used forward-looking viewpoint (from Caltech dataset [34]). Right: example frame from our own recorded dataset.

presented in this dissertation in section 1.1, and give a comprehensive overview of the outline of this dissertation in section 1.2.

1.1 Main contributions

The main contributions that we propose throughout the different chapters in this dissertation can be summarised as follows:

- We give an extensive and thorough overview of the blind spot problem. Detailed information on blind spot accidents and related statistics are evaluated and discussed. Existing systems are analysed, revealing their benefits and disadvantages.
- We propose a generic framework to detect and track vulnerable road users in these blind spot images. Existing pedestrian detectors fail to achieve good accuracy results on these images. For this, we propose our *warping window approach*. Initially, we present a basic framework and then further refine it in consecutive chapters throughout this entire dissertation. We proof the generalisability of this approach by applying a similar detection scheme to a surveillance application.
- To evaluate our algorithms we recorded and labelled unique and valuable datasets in which several dangerous blind spot scenarios were realistically

simulated. These datasets were recorded with a genuine blind spot camera and a real truck, and with multiple *object* classes (such as pedestrians and bicyclists).

- All algorithms proposed in this dissertation were developed for maximal accuracy keeping low computation complexity in mind. To prove this, we developed a real-time demonstrator of one of our algorithms on an embedded platform using a multi-threaded CPU and GPU implementation.
- To improve the detection accuracy of available pedestrian detection algorithms, we propose a methodology to combine several pedestrian detectors in an efficient way. Our experiments indicate a significant increase in accuracy.
- Finally, starting from our VRU tracking framework we developed a complete safety system and performed extensive experiments at system level to verify the usability of this system in practice.

1.2 Outline of this dissertation

To situate and motivate the research conducted in this dissertation, we first give an overview of the blind spot problem in **chapter 2**. We define the actual blind spot zone and give detailed statistics on blind spot accidents and number of casualties. We discuss currently used technical and non-technical solutions. For this, we analyse existing blind spot safety systems, and discuss their advantages and disadvantages. Next we motivate why an active safety system should be developed and define the required specifications that such system should achieve to be usable in practice. Furthermore we discuss intrinsic and extrinsic challenges involved in the detection of VRUs in the blind spot zone when using only a monocular camera as sensor. Evidently, the vision processing pipeline highly depends on the exact camera position. Based on the given statistics we determine the optimal camera position.

We discuss related work in **chapter 3**. Here we focus on the computer vision aspect of this work. We discuss existing pedestrian detection methodologies, and object detection in general. We give an overview of the state-of-the-art and show how the accuracy of different object detection methods is measured and compared.

In **chapter 4** we present an initial approach towards such a complete active safety system. In a first part we present a pedestrian detection and tracking framework able to efficiently detect pedestrians in the typical backward-looking images taken from a regular moving vehicle. In the next part we present a

methodology coined the *warping window approach*. This approach allows for the accurate and real-time detection of pedestrians in these specific blind spot images, where traditional pedestrian detectors would otherwise fail or would be too time consuming. We present a real-time demonstrator of this algorithm on an embedded platform. To achieve real-time performance, we maximally exploit the hardware using a hybrid multi-threaded CPU and GPU implementation.

In **chapter 5**, we prove that the approach presented in chapter 4 is easily generalisable to other application domains. For this, we apply our approach to a surveillance scenario. We exploit the fact that typical surveillance scenarios always employ fixed cameras, and integrate a background estimation technique with our previous approach. We propose experimental results on a publicly available challenging surveillance dataset.

To improve the detection accuracy of pedestrian detectors, we propose an efficient combination methodology in **chapter 6**. This methodology allows for the combination of an arbitrary number of pedestrian detectors based on two measures: the *confidence* of a detector and the *complementarity* of this detector with respect to the other detectors. These two measures are used in a weighted sum to assign a new detection score to each detection. Our experiments indicate that this combination methodology is able to significantly increase the accuracy. Furthermore we show that, given our combination rule and a specific combination of two pedestrian detectors, our methodology manages to find the optimal combination.

To further increase the detection accuracy of our initial approach, in **chapter 7** we extend this framework to cope with the perspective distortion induced by the specific viewpoint. We investigate if the application of a similar strategy as discussed above (combining several pedestrian detectors) could further increase the detection accuracy. Furthermore we extend the framework to cope with an additional important object class – bicyclists – using a probabilistic combination methodology.

We elevate our pedestrian detection and tracking framework to a complete safety system in **chapter 8**. For this, we first present a method to make the calculation time more deterministic and extend the framework to include the detection of children. Next, we propose our final active alarm system, and give extensive accuracy experiments at safety system level. We compare the final achieved specifications of our active alarm system with the required specifications given in chapter 2, and evaluate if our system is usable in real-life scenarios.

Finally, in **chapter 9** we conclude this dissertation with a summarisation of our work, and give an overview of possible future improvements.

Chapter 2

The blind spot problem

In this chapter we aim to give an introduction of the blind spot problem. We first start with a definition of the actual blind spot zones, and indicate how the majority of these blind spot accidents occur. We give statistics concerning the number of casualties, and discuss both technical and non-technical solutions that are currently being used. Next we motivate why we believe a monocular automatic camera-based safety system could be a solution to this problem.

Evidently, the usability of an active safety system highly depends on the specifications that such a system achieves (e.g. with respect to the false alarm rate). We discuss all relevant specifications and give quantitative figures that should be achieved for such a system to be usable in practice.

We then give an overview of the challenges that we need to overcome when developing such an active safety system which only relies on a monocular camera as input. Finally, based on the statistics we discuss how we determined the optimal position of the blind spot camera as used in our experiments.

2.1 Introduction

Each year traffic accidents caused by the blind spot zone of trucks are responsible for an estimate of about 1300 casualties in Europe alone [41]. Since only accidents involving victims are reported and the fact that the exact definition of a blind spot accident includes only very specific scenarios this figure is a great underestimation of the real problem. Several non-technical solutions, such as infrastructural measures or educational programs, are commonly used

today. Despite these actions, the number of severely injured and deaths remains too high (see section 2.3). Therefore, throughout the years several commercial technical systems have been developed that try to cope with this problem. These systems range from simple mechanical solutions (e.g. blind spot mirrors) to more advanced automatic alarm systems, which generate an active warning for the truck driver or the vulnerable road users. However, none of these systems seem able to adequately decrease the number of victims. For example, research indicates that the number of casualties did not decrease since the use of blind spot mirrors was obliged by law in 2003 in Europe [64]. Several other technical systems (e.g. ultrasonic sensors) were already evaluated for their applicability in real-life situations [89]. This study showed that these systems currently are not usable on a large scale due to several false positive alarms, or inadequate/false interpretation of the alarm signals. In this dissertation we therefore aim to overcome the technical challenges in the development of an active safety system using a commercially available blind spot camera as input.

The remainder of this chapter is structured as follows. We first define the blind spot zone and how typical blind spot accidents occur in section 2.2. In section 2.3 we then give exact statistics concerning these blind spot accidents. We compare currently used existing solutions, discuss their advantages and disadvantages and motivate why an active safety system is needed in section 2.4. Next, in section 2.5 we discuss the requirements that such an active safety system should achieve to be used in practice. However, developing such a system based solely on the input of the blind spot camera is challenging, as will be discussed in section 2.6. In section 2.7 we determine the optimal position of the blind spot camera used in our experiments. Finally, we give acknowledgements in section 2.8 and conclude this chapter in section 2.9.

2.2 Defining a blind spot accident

The blind spot zones effectively indicate all zones around the truck (often defined in the literature as *heavy goods vehicle* or HGVs) where the truck driver has no or limited view. The exact definition that is used often depends on the type of statistical data that is available, or on the specific problem that is being discussed. The term HGV indicates all vehicles constructed for the transportation of goods with a maximum mass higher than 3.5 tonnes [90]. In essence there are four zones around the truck which the truck driver is unable to see when using traditional mirrors. These zones are found in front of the truck cabin (to be exact, here the view is limited in height), to the right side of the truck starting from the cabin, behind the truck and at the left-hand side of the truck. Each of these four zones contain one or multiple blind spot zones [2].

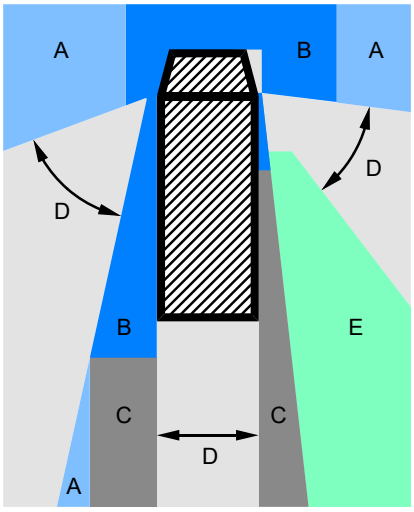


Figure 2.1: Overview of the different zones around the truck. Image based on [115]. Top view of truck (black). (A) Complete view. (B) Limited view in height. (C) View covered with main mirrors. (D) No view. (E) View with wide-angle mirrors (distorted).

The exact positions of these blind spots evidently depend on the type of truck itself. We refer to figure 2.1 for a qualitative overview of the different zones around the truck. As seen, they are subdivided into five categories depending on their visibility with respect to the truck driver [115]: complete view (A), limited view in height (B), view covered with standard main mirrors (C), no view (D) and view with wide-angle mirrors (E). Keep in mind that, although zone (E) should be visible to the truck driver using wide-angle mirrors, this view remains distorted (i.e. estimating distances is difficult).

A traditionally used solution to cope with this problem is the use of special mirrors. Several types of these mirrors exist. Each of them employs a wide angle mirror that, using image deformation, allows to visualise a specific zone. Since 2003, blind spot mirrors covering specific zones were obliged by law in several European countries, including Belgium and the Netherlands. Although these mirrors were able to decrease the blind spot zone, a blind spot remained at the front right side of the truck. Therefore, since 2007 the European Union obliged an additional mirror covering this area as well. See figure 2.2 for an overview of the coverage of the different blind spot zones with specific mirrors. We refer to subsection 2.4.2 for more details concerning these blind spot mirrors. However, despite the obliged introduction of these blind spot mirrors, the

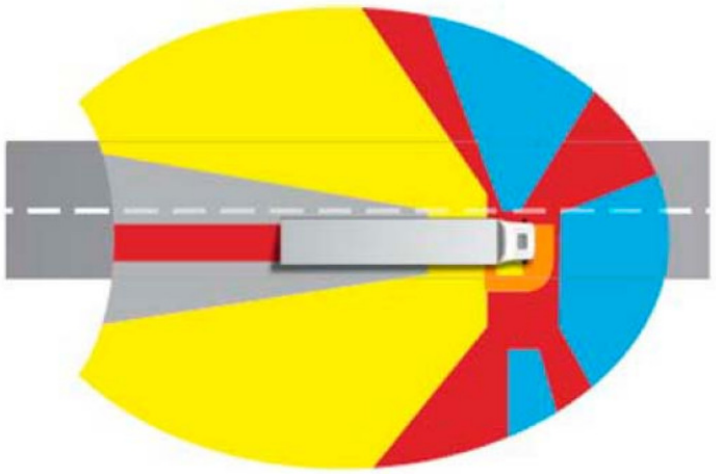


Figure 2.2: An overview of the coverage of the different blind spot zones with specific mirrors. Image from [89]. Top view of truck. **Blue**: direct eye contact with other road users is possible. **Grey**: area must be visible with main mirrors. **Yellow**: Area must be visible with wide-angle mirrors and sidewalk mirrors. **Orange**: area must be visible with frontal mirrors and sidewalk mirrors. **Red**: invisible areas for truck driver.

number of casualties did not decrease significantly [20, 64, 89]. Several possible explanations for this phenomenon exist. These blind spot mirrors need to be adjusted correctly. Research indicates that sometimes these mirrors are deliberately adjusted incorrectly to facilitate manoeuvring. Sometimes the position of these mirrors needs to be readjusted, for example after hitting branches while driving through small streets. Such adjustments are often made incorrectly. To facilitate the correct adjustment of these mirrors, Flanders has several official locations that allow for an exact recalibration. Furthermore, each blind spot mirror introduces an additional new blind spot. However, since nowadays most of the dangerous areas are visible with standard and wide angle mirrors, the responsibility to avoid these accidents has moved more towards the truck driver himself [101]. As such, it is the responsibility of the truck driver to prevent these dangerous situations from occurring. Evidently it is impossible for the truck driver to simultaneously watch through all windows, use all mirrors and look at the monitor of the blind spot camera (if available).

Thus, sadly vulnerable road users are still often the victim of these blind spot accidents. In fact, a questionnaire among truck drivers revealed that only 10% of all truck drivers could identify all problem zones around their vehicle [115].

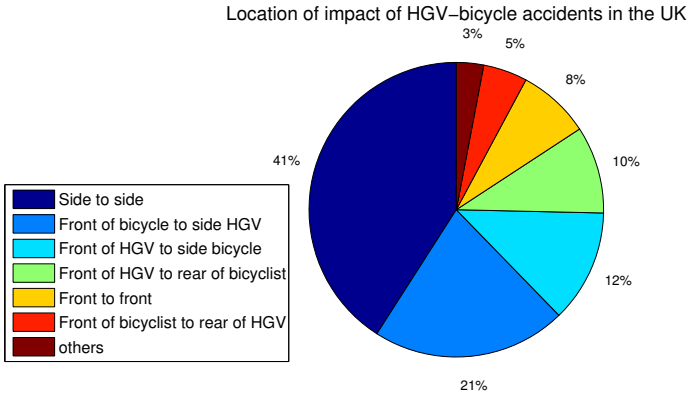


Figure 2.3: Impact location of accidents between HGVs and bicyclists. Statistical data retrieved from [90].

The same research indicates that 78% of all truck drivers has been surprised by traffic in the blind spot zones on multiple occasions.

A blind spot accident is defined as a collision between vulnerable road users and trucks due the blind spot zones. These vulnerable road users consist of pedestrians, bicyclists and mopeds. Often statistics also include motorcycles which are then grouped with mopeds as *two wheeled motor vehicles* (TWMV). Obtaining exact details about the circumstances of blind spot accidents in Belgium is difficult, since these statistics are not available. However, in 2010 the *Transport Research Laboratory* (TRL) published extensive research results concerning heavy vehicle crash statistics [90] in Great Britain. Using these statistics we were able to segment the different accidents between HGVs and bicyclists, see figure 2.3. This figure displays the impact location of accidents between bicyclists and HGVs measured over three years (2006 – 2008). These statistics include both severely injured and fatally injured.

A detailed look at these statistics reveals that the majority of these accidents (the top three, 74% in total) involve collisions where the truck hits the side of the bicyclist either with its front or side, or where the bicyclist hits the front of the HGV. Indeed, this is the most typical scenario: the truck driver makes a right turn at low speeds (or a left turn in left-driving countries), while the bicyclist continues its way straight ahead [56]. The truck driver is not aware of the presence of the bicyclist, and a collision occurs at the front right (or left in left-driving countries) of the truck [95]. Often these bicyclists get caught under the front wheels, resulting in serious injuries or even death. Concerning pedestrian casualties the report indicates that about 51% are hit by the front of the truck, 37% by the side of the truck and 12% by the rear-end of the truck.

Therefore, in the remainder of this dissertation we employ the term *blind spot zone* to indicate the front right side of the truck, and we classify *blind spot accidents* as vulnerable road users which are (fatally) injured after a collision with the truck in this specific zone. In the next section we give detailed statistics concerning the number of casualties and how they are divided over the different vulnerable road users.

2.3 Statistics of blind spot accidents

Research indicates that trucks cause relatively few accidents compared to the travelled distance [2]. However, the consequences of such an accident are much more severe. Indeed, for every 1000 accidents involving casualties in Belgium, about 25 deaths are counted. However, if only accidents with trucks are included, this number increases to about 60 deaths [11]. When compared to their relative presence on the road, significantly less blind spot accidents occur between bicyclists and cars as between bicyclist and trucks. Exact statistics concerning blind spot accidents in Belgium are unavailable. To get an estimate of this figure, a blind spot accident is approximated with the following conditions (as used by the Belgian Road Safety Institute (BIVV) to extract the statistical data):

- The truck takes a right-hand turn
- A pedestrian, bicyclist or moped is the victim
- The road on which is being driven is equal for both road users
- The direction of movement is equal for both road users

Since these conditions are quite strict these figures exclude blind spot accidents occurring when e.g. the truck takes a left-hand turn and blind spot accidents that are caused by the front and rear blind spot zone – they are thus an underestimation of the number of casualties. The resulting statistics from 1998 – 2007 are visualised in figure 2.4 [71, 89].

As can be seen, over this period neither the number of accidents nor the number of casualties showed a significant decrease. About 60 – 70 blind spot accidents occurred each year, resulting in 40 – 45 slightly injured, 10 – 15 severely injured and about 8 people who died yearly in a period of 30 days after the accident occurred. To compare these figures with all traffic casualties in Belgium, over the past five years (2011 - 2015) on average about 43000 traffic accidents occurred, resulting in about 57000 victims of which 760 victims died. These statistics were

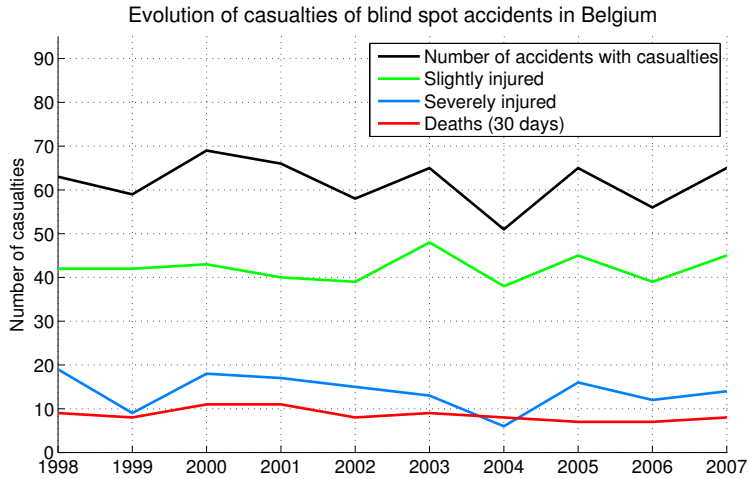


Figure 2.4: Evolution of the casualties involved in blind spot accidents in Belgium ranging from 1998 – 2007. Statistical data retrieved from [71, 89].

retrieved directly from the Belgian Federal Government (Statbel). However, even if the number of casualties due to blind spot accidents is small in absolute figures, it remains a relevant problem which needs to be tackled. As previously mentioned, the obliged introduction of the blind spot mirror in 2003 did not result in a durable significant decrease in the number of casualties.

Since 2008 the BIVV avoids the distinction between *slightly injured* and *severely injured* since this subdivision proves to be inaccurate. In new reports they are gathered under the same denominator (*injuries*). Figure 2.5 shows the most recent available reliable statistics at the time of writing of this dissertation (April 2016). A decrease in the number of casualties can be observed starting from 2009. A possible explanation for this could be found in the introduction of the additional blind spot mirror in 2007. It is evident that this introduction is only noticeable a few years later as only new trucks employ these mirrors – no retrofitting was obliged.

In 2011 there even were no deaths counted in Belgium. However, the number of injuries again increased. Starting from 2011 again an increase in casualties can be observed indicating the decrease observed earlier could not be confirmed. Analysis concerning the type of casualties showed that in Belgium, most of them (combining both injuries and deaths) are found among bicyclists (68%), followed by TWMVs (30%). Only about 2% of these casualties are pedestrians.

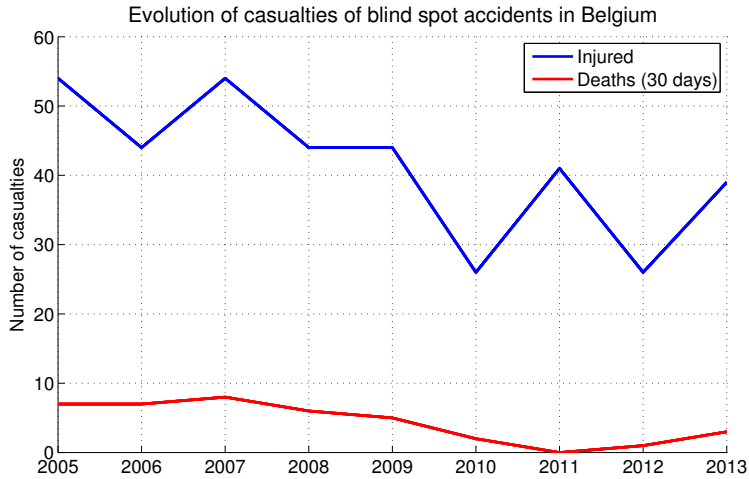


Figure 2.5: Evolution of the casualties involved in blind spot accidents in Belgium ranging from 2005 – 2013. Statistical data retrieved directly from the BIVV in April 2016.

A similar observation is seen for the Netherlands. No explicit data about the subdivision between VRU classes is available since they experience that most casualties are found under bicyclists and thus this class is given more attention. On average, about 14 bicyclists are killed each year in blind spot accidents. No long-term decrease has been noticed over the period ranging from 1997 – 2009. Around 2002 and 2003 the number of deaths were lower (6 and 7 respectively), mainly due to the publicity involved around the introduction of the blind spot mirror. This decrease was only temporary [95].

However, a comparison with other countries in Europe reveals that bicyclists being the main victim is more of an exception. For example, in Great Britain an annual average of 72 pedestrians, 27 bicyclists and 34 TWMVs fatalities are counted due to accidents involving HGVs, while 151 pedestrians, 72 bicyclists and about 110 TWMVs were seriously injured [90]. When estimating only the share of blind spot accidents (based on the impact location) they are almost equally divided: about 75 pedestrians (34%), 72 bicyclists (33%) and 73 TWMVs (33%) were involved (including both seriously injured and fatalities). In France, 12 bicyclists and 6 pedestrians were killed in 2008 [56]. In Germany 135 accidents between HGVs and VRUs occurred in 2002, resulting in 10 deaths [75].

Based on the statistics discussed above, it is clear that the blind spot zone each year remains responsible for a large number of casualties and severely injured.

To cope with this problem, several technical and non-technical solutions have been proposed in the past. In the next section we analyse each of these existing solutions. As will be discussed, an optimal solution currently does not exist.

2.4 Existing blind spot solutions

In this section we now discuss existing solutions targeting the blind spot problem. These are subdivided into two categories: non-technical and technical solutions. We discuss both categories and analyse the advantages and disadvantages of the existing technical blind spot systems. Furthermore we discuss research results obtained by the BIVV where large-scale tests were conducted with two commercially available technical blind spot detection systems. We show that none of them seems to cope with the problem completely. Finally, we conclude this section with a summarisation of the main advantages of an active camera-based blind spot detection system, and motivate why we believe a monocular automatic camera-based safety system could be a (partial) solution to this problem.

2.4.1 Non-technical blind spot solutions

For the sake of completeness, in this subsection we give a concise overview of currently used non-technical measures. For more details we refer to [89, 95].

Education and sensibilisation

Evidently, all road users need to be informed about the blind spot problem. The most dangerous zones, and most typical scenarios ought to be known by both VRUs and HGV drivers, which is not always the case in practice. Although most responsibility is on the HGV driver (since VRUs often have immediate priority in most blind spot scenarios) the VRUs need to be able to use their priority in a sensible manner: e.g. maintaining eye contact with the truck driver, avoiding the blind spot zone in general or avoiding to stop directly under the mirrors. Furthermore, HGV drivers need to be reminded to use correct safety procedures: e.g. checking all mirrors when making a right-hand turn and making sure that all mirrors are always adjusted correctly.

Infrastructure

These measures aim to avoid dangerous blind spot situations altogether. This is done in several different ways, ranging from short term measures (such as the construction of directional lanes for bicyclist or the adjustment of traffic light phases) to more structural measures which limit the simultaneous presence of both VRUs and HGVs on the same road. These infrastructural measures are in fact the most recommended strategic solution according to *SWOV* (the national research institute concerning traffic safety in the Netherlands) [95].

Constructional measures

The exact blind spot zone depends on the specific type of HGV. For example, the size of the cabin and front window could be optimised to minimise the blind spot zone. Furthermore, lateral protection systems (side guards) are often a mandatory requirement on new vehicles. These side guards aim to prevent that bicyclists end up under the wheels of the truck.

2.4.2 Technical blind spot solutions

In this subsection we give an overview of existing technical blind spot systems: blind spot mirrors, passive camera systems, ultrasonic sensors and VRU warning systems. Indeed, several such systems exist relying on different underlying technologies. These technical blind spot solutions are again subdivided into two different types: *active* systems and *passive* systems. Active safety systems automatically generate an alarm if VRUs are present in dangerous zones around the truck (e.g. ultrasonic distance sensors), whereas passive safety systems still rely on the focus of the truck driver (e.g. blind spot mirrors). However, each of them has specific advantages and disadvantages. As of yet, a perfect technical VRU detection system is nonexistent.

Blind spot mirrors

Evidently, mirrors are one of the traditionally used approaches to cope with these blind spot zones. They are relatively cheap compared to other detection systems. However, they are far from an ideal solution. Figure 2.6 gives an overview of all possible mirrors found on the right-hand side of a HGV, of which some are currently enforced by law. In this subsection we only focus on the mirror obligations in Belgium. Similar observations are seen for other European

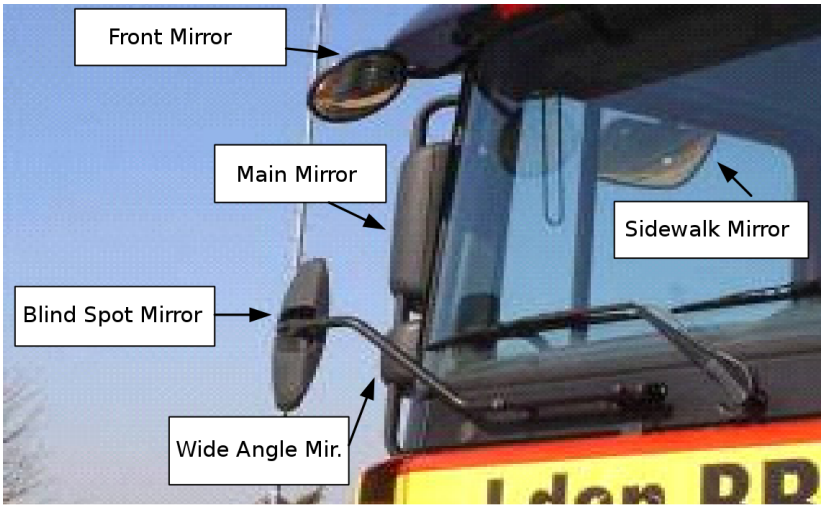


Figure 2.6: An overview of all possible mirrors available on the front right side of a HGV. Specific mirrors are obliged by European laws. Image from [95].

countries. Apart from the main mirrors, since January 1991 new trucks were obliged to install two additional mirrors: the sidewalk mirror and wide angle mirror. Although these two mirrors decreased the size of the blind spot zone, a significant part was still not covered. Therefore, since January 2003 new trucks were enforced to include an additional blind spot mirror, making a total of four mirrors to cover the right-hand side of the truck.

Evidently, all these mirrors made it difficult to form a clear overview, and it was unfeasible to utilise all of them simultaneously. To overcome these problems the European Union enforced a specific law in January 2007 in which they state all areas that need to be visible to the truck driver. To meet these requirements a front mirror should be installed and the viewing angle of both the sidewalk mirror and the wide angle mirror need to be increased, making the blind spot mirror redundant. Despite all these mirrors, section 2.3 indicated that the number of casualties did not decrease. This is mainly due to the fact that these mirrors are not used or adjusted correctly. In fact, these mirrors are often deliberately adjusted incorrectly to facilitate manoeuvring. It remains a passive system which the truck driver still needs to use in an efficient manner. Viewing all mirrors at the same time remains a difficult task. Furthermore each new mirror implies an additional obstruction in the field of view of the driver, and thus an additional blind spot. Additionally, due to the view distortion it often is difficult to correctly estimate distances.

Passive Camera Systems

A step up from traditional mirrors are found in passive camera systems. Here, when the truck driver signals a right-hand turn, a monitor in the truck's cabin is switched on that displays the blind spot camera image and thus gives a clear overview of the blind spot zone. This system has several advantages: it is mounted in a robust manner such that future adjustments are not needed.

Furthermore such camera systems are small and thus have no impact on the view of the truck driver. They can be mounted at any location, giving an excellent overview of dangerous blind spot zones. This is a significant advantage over the standard blind spot mirrors, especially for semi-trailer trucks. Due to the semi-trailer, during a right-hand turn of these vehicles the fixed viewpoint of the mirrors no longer covers the actual blind spot zone. Many cameras additionally use infrared lighting such that they can be used in low-vision circumstances.

Recently, a new passive camera system was made commercially available, called *Omni-Vue*. Here, four cameras are mounted around the truck – one at each side. The images from all four cameras are warped and stitched together, and the truck driver is presented with a bird's eye view around the vehicle.

However, these safety systems remain passive and as such have similar disadvantages as compared to the standard blind spot mirrors.

Ultrasonic distance sensors

As mentioned, ultrasonic distance sensors are a type of safety systems that are coined active systems. They automatically generate an audible or haptic signal to the truck driver when an object is detected. These systems consist of transceivers on the right-hand side and front side of the truck and, based on the reflections, they measure if an object is in close proximity to the vehicle. The detection range of these systems is often around 1 to 1.5 metres, and they are activated below certain speeds and when the turn indicator is used.

Evidently, the main advantage of these systems is found in their active alarm indication such that the driver's attention is automatically drawn. These systems however are unable to distinguish true vulnerable road users from nearby static objects such as traffic signs or trees, and thus often generate unnecessary alarms (called *false positives*). Research shows (see subsection 2.4.3) that the truck drivers find these signals annoying and therefore tend to completely disable these systems rendering them useless. Also, the use of these camera and/or ultrasonic systems do not eliminate the obligations with respect to the mirror systems.

VRU warning systems

A final type of safety systems act as alarm system warning the vulnerable road users in close proximity to the truck themselves. They are not used to increase the field of view of the truck driver, but rather tend to raise awareness to the surroundings of the truck for potential dangerous situations. These systems are activated when the turn signal is enabled below a certain speed. They generate an audible and visual signal indicating a right-hand turn using warning lights mounted on the exterior of the truck.

A main disadvantage of these systems is found in the interpretation of these signals by the vulnerable road users. They must interpret it correctly and act in time to avoid an accident. Furthermore if only a limited amount of trucks is equipped with such a system, the vulnerable road users are unable to rely on these alarm signals since no alarm would then not automatically indicate a safe situation. And since the truck driver assumes that the VRUs are warned in advance, dangerous situations could occur if he blindly relies on the system. Additionally, such systems cause noise pollution towards local residents.

2.4.3 A study on the effectiveness of these technical solutions

In the subsection above we discussed current available technical blind spot safety systems. These systems are on the market for a substantial amount of time. However, none of them seems to be able to decrease the number of casualties. An important comparative study was conducted and published by the BIVV in 2011 [89], which was commissioned by the Belgian Government. In this research project, seven commercial technical systems were evaluated, and two of these systems were selected to be employed in a larger study: the *Lexguard* system and the *Lisa-2-Alert* system. The first system relies on ultrasonic distance sensors measuring distances up to 1 metre. The truck drivers were unable to deactivate the system. The second system is a VRU warning system that indicates a right-hand turn as mentioned above, using a continuous audible signal (80 dB at 5 metres away) together with blinking lights at the side of the truck. Since this system is not in accordance with official Belgian laws, an exception was made and the truck drivers were able to deactivate this system. See figure 2.7 for an overview of both systems.

The initial goal was to mount each of these systems to 50 trucks, for a period of three months. In practice, 33 trucks were equipped with the Lisa-2-Alert systems, and 38 with the Lexguard system. A control group of 48 trucks was used to test the effect of the technical systems. This was needed since the behaviour of the participating truck drivers might have changed due to the

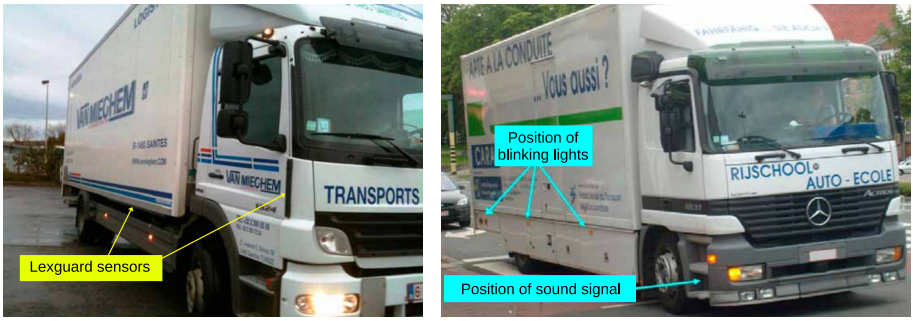


Figure 2.7: An overview of the Lexguard system (left image) and the Lisa-2-Alert system (right image). Images from [89].

inherent sensibilisation. Each truck driver needed to maintain a logbook for each day in which he noted down blind spot situations. Several surveys were conducted among these truck drivers and bicyclists to observe their perception of the Lisa-2-Alert system.

They concluded that the Lexguard system in its current form was unpractical to use due to two reasons. Many false positive alarms (especially during rainy conditions or even snow) occurred, and an alarm is given too late since the sensors are mounted on the side of the truck. An alarm should be given in time to cope with the reaction time of the driver. The Lisa-2-Alert system was positively received. However, the warning signal was often incorrectly interpreted (only 28.6% understood the signal) or not always noticed (e.g. when wearing headphones – 25.6% did not hear the signal).

2.4.4 Conclusion

As discussed above a perfect safety system currently does not exist. An active safety system based on the blind spot camera images has several advantages as compared to the previously discussed systems. Such a system is always adjusted correctly, and is easy to implement in existing passive blind spot camera setups. Since the detection of vulnerable road users is done completely automatically, no interpretation by the truck driver is needed. Due to the specific viewpoint of the camera an alarm can be given if the bicyclist enters the blind spot zone at the rear of the truck, leaving enough time for the truck driver to react. Furthermore, since we aim to design an accurate detection algorithm, this system is able to distinguish pedestrians and bicyclists from e.g. traffic signs or other static objects. This should significantly reduce the number of false positive alarms.

In the next section we analyse the requirements that such an active alarm system should achieve, and as such present the required design specifications.

2.5 Specifications of an active alarm system

The acceptance and effectiveness of an active alarm system highly depends on the specifications that such a system achieves. For example, if the false alarm rate proves to be too high, the truck driver will experience these alarms as annoying and tends to ignore – or even completely disables – the system. In this section we therefore discuss the most important criteria, and give quantitative figures that such a system should achieve in order to be usable in practice.

A commercial active blind spot safety system consists of at least two components: a detection component for the vulnerable road users and a component that informs or warns the truck driver of their presences. More autonomous systems might include additional actions such as automatic braking manoeuvres. The effectiveness of such as system depends on several factors such as the usability (e.g. number of false alarms and ease of use), reliability (e.g. the down time), the social acceptability, cost and so on [56]. Additionally, the predictability of such a system is crucial. If the situations in which false alarms occur is predictable (e.g. during rain), a higher false alarm rate might be accepted as opposed to when the alarm system generates false alarms at random moments.

Several important criteria exist. For example, such a system should not increase the workload of the truck driver and the information must be conveyed in such a way that the truck driver is informed in a clear manner while not being distracted. If the system imposes additional automatic actions, these should only be used in very specific time-critical moments. Therefore, such as system should be able to make an optimal distinction between critical and non-critical situations.

In this dissertation we only focus on the detection component of the system. Since we do not aim to deliver a final commercial product, no research was performed concerning the interaction between the truck driver and the system. Here, we thus only discuss the technical specifications that such a system should achieve with respect to the detection of vulnerable road users to be usable in practice. According to [56], two main technical criteria exist: the system should be able to perform *good* VRU detection and the system should be able to give an alarm system *in time* such that enough time remains for the truck driver to take action. We translated both criteria in three distinctive requirements: the *throughput*, *latency* and *accuracy* of the system. We now discuss each of the following criteria in more detail, and give quantitative measures for these

criteria. In chapter 8, section 8.5 we will compare the specifications that our final alarm system achieves with the requirements discussed here, and discuss the usability of our final alarm system in real-life situations.

2.5.1 Throughput

The throughput is defined as the productivity of a system. The throughput indicates, given a specific amount of time, how much information can be processed within that time. This definition is applied to our specific application as follows. Our blind spot camera captures new images at discrete moments in time, expressed as the number of frames per second. A commercial blind spot camera – including the one used in our experiments as indicated further – achieves a frame rate of 15 frames per second. Evidently, our application relies on *hard real-time* operation. A system is seen as real-time when the task which it performs is executed in a time frame that is not longer than the maximal time that is allowed. When applied to the image processing context, a system is defined as being real-time when the system is able to process the images not slower than the rate at which new images need to be processed (which then is the allowed time). A system is considered *hard* real-time when the system must always achieve this deadline (i.e. the maximally allowed time). This constraint is stringent: if the deadline is not met even once, the system has failed.

Such hard real-time behaviour is needed for these critical safety applications. Exceeding this deadline might have severe consequences (e.g. resulting in lethal injuries). Thus, for our specific application, our final alarm system must guarantee that each new image frame is processed at least at the rate of new incoming frames. Thus, our system needs to achieve a processing speed (i.e. throughput) of 15 frames per second, the upper bound of the time to process an incoming image is 1/15th of a second.

2.5.2 Latency

The latency is defined as the delay from input to output that a system introduces. In our context, it indicates the time between the moment that VRUs enter the blind spot zone, and the generation of an actual (audible, visual or haptic) alarm signal. In the optimal case, an alarm should be generated immediately when the VRUs enter the blind spot zone. This, however is unfeasible. Keep in mind that in practice the latency is influenced by several factors.

The image frames need to be acquired before they are processed. This is done using *frame grabbers*, which introduce small delays. Furthermore, each frame

needs to be processed to evaluate the presences of VRUs. At processing speeds of e.g. exactly 15 frames per second, this delay equals 67 ms.

To determine the maximally allowed latency, several non-technical factors need to be taken into account. Research concerning the reaction time among truck drivers when confronted with an event (e.g. hearing an alarm signal) and the effective brake action reports times of about 1.5 seconds [19]. However, this reaction time depends on the age of the truck driver, and the confidence of the driver with respect to the alarm system. If the driver is more familiar with the system and the meaning of the alarm signals, this reaction time decreases.

Furthermore, after the brake action is performed, time is needed to perform the stopping manoeuvre of the truck. The required time to perform this manoeuvre again depends on several external factors, such as e.g. the weather conditions. Our system should be able to warn the truck driver at least this combined time in advance. To achieve this, the maximal latency that our final alarm detection system requires depends on the relative speed difference between the VRUs and the truck, and the size of the blind spot zone in which detection is performed. Indeed, if we are able to detect the VRUs already far behind the front of the truck, the latency requirements become less stringent. Since the exact latency requirement depends on many factors and parameters, giving a single final figure is difficult. We refer to chapter 8, section 8.5 for a detailed discussion on this matter.

2.5.3 Accuracy

As discussed above, the system should be able to perform *good* VRU detection. As such, the false alarm rate and miss rate should be minimal. Exact quantitative figures of these false alarm rates are difficult to find and not consistent in the literature. For example, research on vehicle-based pedestrian collision warning systems state a maximum false alarm rate of 2% and a miss rate of 1% [15] need to be achieved, whereas [56] indicates a false alarm rate of up to 5% is allowed.

Defining exact figures is difficult, since these requirements highly depend on how these particular safety systems are used. For example, if used on an autonomous driving vehicle, perfect accuracy is acquired. As such, no false alarms or missed detections are allowed. However, if used as decision support safety system (where the truck driver remains responsible) this is not the case. In such cases, non-zero miss rates are allowed when no (or only few) false alarms are given. For example, if a system is able to detect e.g. 80% of all VRUs in the blind spot zone while ensuring no false alarms are given this system is of great help. It is still the driver's responsibility to look at the blind spot camera monitor

during each manoeuvre to check for VRUs that the automatic alarm system could miss. Furthermore, as mentioned low false alarm rates are accepted if their occurrence is predictable, but of course must be minimised as much as possible.

2.5.4 Conclusion

In this section we discussed the specifications that an active blind spot alarm system should achieve in order to be usable in practice. For this, we defined three measures and discussed the requirements for each of them. In chapter 8, section 8.5 we compare our final alarm system developed throughout this dissertation with the requirements given here, and discuss the applicability of our final system towards real-life application.

However, developing such an active safety system based solely on the input images from the blind spot camera is a difficult task, as discussed in the next section.

2.6 Challenges in the development of a camera-based active safety system

In this section we aim to give an overview of the main challenges that we need to tackle in order to develop our active alarm system based on the blind spot camera images. Several of these challenges will be addressed in the different chapters of this dissertation.

Firstly, as mentioned above, such a safety system needs to achieve high detection accuracy. To achieve such high detection accuracy demands, often efficient implementations on specialised hardware (such as multicore CPU and high-end GPUs) are required. Aside from high accuracy, this specific application also implicitly requires *real-time* performance: each incoming blind spot image needs to be evaluated in a very limited time-frame.

These two demands are often contradictory: achieving high accuracy often comes at the cost of high computational complexity. Furthermore, the safety system that we aim to develop needs to be implementable in HGVs in real-life scenarios. Hence, it is impossible to rely on high-end hardware for this application, such as computer clusters. In fact, we target algorithms which are able to achieve real-time performance on small, light-weight embedded platforms at high accuracy. Most algorithms presented in this dissertation were developed

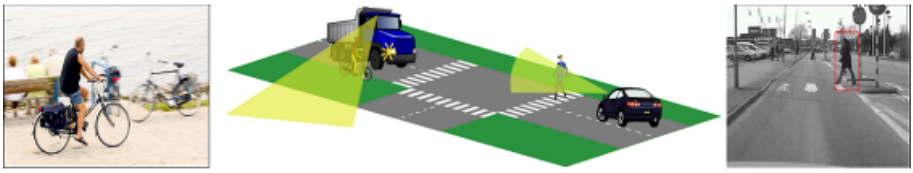


Figure 2.8: Image indicating the difference in viewpoint between the traditionally used forward-looking cameras and the camera viewpoint used in our application.

with this idea in mind. For example in chapter 4, section 4.3 we present such an efficient hybrid implementation which maximally exploits both the CPU and GPU capabilities on a Mini-ITX embedded platform resulting in real-time performance.

Secondly, our camera is mounted on a moving vehicle. Therefore we need to evaluate if motion blur could be an issue. Since the background is highly dynamic, we are unable to employ often used background estimation techniques as a preprocessing step to speedup detection. For this, we need to rely on appearance-based techniques. Furthermore, due to the inherent wide-angle lens of the blind spot camera, we need to cope with the severe distortion and specific viewing direction. This is one of the major differences with existing car safety systems (such as the *Volvo* and *Mercedes* pedestrian detection system), which often rely on frontal looking camera. Figure 2.8 qualitatively displays the difference in viewpoint. Such forward-looking viewpoint significantly simplifies the detection problem.

Finally, several challenges arise from the specific *objects* that we aim to detect. Apart from pedestrians also bicyclists, mopeds, children and wheelchair users are considered vulnerable road users. See figure 2.9 for an overview. Some of these classes (e.g. bicyclists) have a very distinctive view based on the angle from which they are seen, called the *multi-view* problem. Furthermore their appearance varies significantly (e.g. different clothes, backpacks and so on), and they are *non-rigid objects*: the exact pose of a walking pedestrian strongly differs from stationary pedestrians.

In this dissertation we extend our detection framework throughout the different chapters to gradually include more VRU classes. We first develop an initial pedestrian detection and tracking framework for these challenging blind spot images in chapter 4, which we then extend to bicyclists (chapter 7) and finally include children (chapter 8). In the next chapter we now discuss how we determined the optimal location where the blind spot camera should be mounted on the truck.

Figure 2.9: Vulnerable road users consist of very diverse classes. Aside from pedestrians, also bicyclists, mopeds, children and even wheelchair users are included.

2.7 Determining the optimal blind spot camera position

Evidently, the position of the blind spot camera on the truck has a large impact on the consecutive vision processing pipeline. In the remainder of this dissertation we employ this vision sensor as our only input. Thus, we do not rely on e.g. distance sensors or infrared images. Our motivation for this is two fold. First, we aim to develop an active safety system which is easily integratable in existing blind spot camera solutions. Such passive camera systems are already widely spread among transport companies. Second, this dissertation aims to explore the accuracy limits when using only a vision-based sensor. If we eventually conclude that the optimal accuracy we achieve is not suitable for real-life situations, the inclusion of other sensors can be seen as future work. Indeed, these additional sensors could for example be used to validate VRU detections from our framework and as such further reduce the number of false positives.

For our vision sensor we thus rely on a genuine, commercially available blind spot camera. Several well-known Flemish companies, such as *Van Dievel Transport* and *Colruyt Group* already employ these wide-angle lens cameras on their entire HGV fleet [36, 116], albeit as a passive camera system. When the truck driver signals a right-hand turn, the camera system is activated. Using a monitor in the truck's cabin the truck driver needs to determine the presence of VRUs in the blind spot zone. In our experiments we employ the Orlaco115° blind spot camera. It has a viewing angle of 115 degrees, and outputs analog VGA resolution images (640×480 pixels) at 15 frames per second. A built-in heater is present to prevent fogging of the lens during winter conditions.

Based on the previously discussed statistics we determined the most optimal camera position as follows. Evidently, the right side is responsible for most blind spot accidents and should be extensively covered. Furthermore most blind spot accidents occur in similar scenarios where a bicyclist drives straight ahead while

Figure 2.10: Our test truck used throughout this dissertation consists of a Volvo FM12. The right image indicates the position of our blind spot camera.

the truck driver makes a right-hand turn. Therefore we opted to install the camera such that the right side of the truck is completely covered starting from the front of the cabin while trying to maximise the viewing distance towards the rear of the truck.

For this, we mounted our camera at the position of the sidewalk mirror. This position allows for an early detection which proves to be crucial to warn the truck driver in time in order to prevent these accidents from occurring. Our test truck, used to record all simulated scenarios in the remainder of this dissertation, is a Volvo FM12. Figure 2.10 displays the exact mounting position of the camera on the vehicle. An example frame as seen from this camera position is visualised in chapter 1, figure 1.2 (page 4).

2.8 Acknowledgements

We would like to acknowledge several people from the BIVV who gave us valuable information concerning the blind spot problem, existing solutions and access to statistical data: Lars Akkermans, Koen Peeters, Stef Willems and Jean-François Gaillet.

2.9 Conclusion

This chapter served as a motivation to why this doctoral research is needed. We extensively discussed the blind spot problem, and gave insights in how these typical blind spot accidents occur. We discussed how the vulnerable road users are distributed among the casualties, and which non-technical measures are currently taken. Furthermore we extensively discussed existing technical blind spot solution systems. We showed that, despite all these existing safety systems and precautions, the number of casualties did not decrease. Therefore, we believe that an active safety system, which automatically warns the HGV driver of the presence of VRUs in the blind spot zone, could help to decrease the number of casualties.

We aim to develop such a safety system which is easily integratable in existing passive blind spot camera setups, by developing computer vision algorithms that are able to detect the VRUs in these blind spot camera images. However, such an active safety system requires stringent specifications to be used in practice. In this chapter we discussed these requirements, and thus set the guidelines for the performance that our final alarm system should achieve.

Furthermore we showed that the detection of these VRUs in the blind spot camera images is far from trivial. Moreover, the exact position of the blind spot camera has a considerable impact on the consecutive computer vision processing pipeline. Based on the blind spot statistics, we determined the most optimal position where the blind spot camera should be mounted on the HGV.

In the next chapter we will give an overview of related work concerning computer vision object detection algorithms which are of interest for the remainder of this dissertation.

Chapter 3

Related work

In the previous chapter we thoroughly discussed the blind spot problem and existing solutions. We showed that, despite the fact that several commercial technical blind spot detection systems are available, a perfect detection system currently is nonexistent. Furthermore we motivated why we believe that a camera-based safety system could be a solution to this problem, and discussed the specifications that such a system should achieve.

Using computer vision detection methodologies, we aim to develop such an automatic alarm system which warns the truck driver of the presence of VRUs in the blind spot zone. Therefore, in this chapter we concentrate on the computer vision aspect of this dissertation and discuss existing object detection methodologies with a main focus on pedestrian detection.

We give a concise overview of the evolution of these detection methodologies, and we discuss techniques that we used and evaluated throughout the different chapters of this dissertation. Additionally, we give an overview of how these different detection methodologies are compared in a fair manner.

Where needed, subsequent chapters in this dissertation have an additional related work section which focuses on related work concerning that chapter's specific content.

3.1 Introduction

In this section we give an overview of related work concerning pedestrian detectors. In section 3.2 we explain how object detection in general is performed, and how these detection models are learned and evaluated. We then give a concise overview of the evolution of pedestrian detection methodologies in section 3.3. A complete detailed explanation of each detector is out of the scope of this chapter – this section mainly serves as a motivation and background to why specific design choices were made in subsequent chapters. Where needed, (e.g. when a specific pedestrian detector is used in our framework) we give a more detailed explanation. In section 3.4 we explain how the accuracy of these pedestrian detectors is evaluated. We conclude this chapter in section 3.5.

3.2 Object detection algorithms

In general object detection is mostly performed as follows. For each input image, a fixed-size model of the object to be detected is evaluated over the entire image. Since the scale of the objects that need to be found is often unknown in advance, a first step consists of the construction of a *scale-space* pyramid. For this, the input image is scaled such that the size of the smallest objects that need to be found in the images matches with the size of the detection model itself (often upscaling is needed for this). This image is then downsampled with a specific factor and smoothed such that larger instances of the object are also detectable. This downsampling is repeated for several scales, until the total image size approximately equals the model size. This downscaling ratio is often given as the number of scales per octave (indicating a factor two in resolution).

For each layer of this scale-space pyramid several *features* are calculated, and thus a *feature pyramid* is constructed. These specific features are detector dependent. Since the position of the objects are again unknown, each position in this entire scale-space needs to be evaluated. Evaluation all scales is often unfeasible to perform in practice. Only specific positions are evaluated using a discrete step size of a few pixels (depending on the scale). The evaluation of each position is often coined as the *sliding window* approach. Evidently, the image size and the number of detection scales has a large impact on the calculation time. Therefore, an obvious technique to reduce the calculation time is to limit the number of scales such that detection accuracy remains sufficient.

At each position that is evaluated, these features are extracted and compared with the pre-computed model of the object. Again, this exact comparison is detector dependent. Often a specific machine learning classifier is employed.

For example, the extracted features could serve as input vector to a *support vector machine* (SVM) – the detection model – which then classifies this image position. Other approaches combine specific features in e.g. *decision trees*. Each of the aforementioned methods often returns a *detection score* for each position, indicating a certain probability score of that image patch as containing the object to be detected. A threshold on this score is used to set a specific *working point* for a detector. Image positions with a score above the threshold are indeed considered to contain the object of interest whereas scores below the threshold are considered to belong to the background.

After this classification step, all possible object locations are indicated with a *bounding box* indicating the scale, position and score of the detections. Due to the sliding window approach, multiple overlapping detections for a single object are often found around the same location. To cope with this, an additional *non-maxima suppression* (NMS) step is performed. Based on an overlap criterion overlapping detections are merged and only the highest scoring detection is kept. This NMS step is more critical than one initially assumes, since two objects which are close to each other (often the case in pedestrian detection) still need to be detected as individual objects while only one detection per object should be found. For a detailed study on the impact of the type of NMS, the overlap threshold and detection accuracy we refer the reader to [33].

The object detection model is learned in advance in an offline step. For this, manually labelled data of positive images (containing the object) and negative images (images without the object) is needed. Several hundreds or even thousands of images are typically used. All features on these images are calculated, and these features then serve as input to the machine learning classifier which aims to identify the features which optimally distinguish between the positive and negative images. Some classifiers simply learn a specific decision boundary in the feature space, whereas others in fact learn which weak features need to be combined to perform object detection (e.g. *AdaBoost*). The main goal of this training stage is to determine a model which is able to generalise well enough to detect instances not seen in the training set but specific enough to only detect the objects of interest. Indeed, detecting specific instances of objects in images remains a challenging task due to changes in their appearance, viewpoint, changing illumination and occlusions.

3.3 Evolution of object detection methodologies

Object detection in general is a challenging task due to the wide variety in both objects and backgrounds. Many computer vision applications (e.g. traffic,

surveillance, industrial automation and so on) rely on efficient object detection algorithms. Specifically, pedestrian detection is a very active research topic for multiple reasons. Accurate pedestrian detection enables numerous important safety applications, the problem is well understood and extensive labelled benchmark datasets and methodologies exist enabling a fair accuracy comparison. Many of the detection methodologies initially developed for pedestrians are easily applied to more general objects. This section aims to give a brief overview concerning the evolution of these pedestrian detection algorithms. For more details, we refer the reader to several extensive comparative works [7, 34, 35, 39, 126].

In 2001, Viola and Jones proposed the use of rectangular *Haar features* (based on [80]) for face detection [117]. These features consist of two, three of four rectangular regions (depending on the feature) and the feature values are calculated as the difference between the sum of the pixels within these regions. These rectangular features are calculated in a very efficient manner – and independent of their size – using an intermediate image representation called *integral images*. The authors employed AdaBoost to select only a small number of important features during the training stage. During detection a fast rejection of negative patches is obtained using a cascaded pipeline of many small boosted classifiers. Only patches that have a positive evaluation at a certain stage are further processed whereas a negative evaluation completely rejects the image patch. In 2003 they extended and applied their previous work on face detection on the task of pedestrian detection [118]. Several extensions of these Haar-like features were proposed later on [70, 74]. In 2004, Ahonen et al. presented the use of *Local Binary Patterns* (LBP) [78] for face recognition [1]. This LBP operator assigns a label to each pixel by thresholding this pixel with its local neighbourhood (e.g. in a 3×3 window). A histogram of these labels is used as texture descriptor.

Dalal and Triggs proposed the use of *Histograms of Oriented Gradients* (HOG) for pedestrian detection in 2005 [21]. Their work clearly outperformed the wavelet-based approach. Here the orientation of the gradients is stored in histogram bins, weighted with the gradient magnitude. Several normalisation steps are performed, and these histograms are then collected over the entire window that is being evaluated, and used as input features for a linear SVM. Since then, both the accuracy and speed of pedestrian detection algorithms steadily increased. However, even today their work is used as a reference in new benchmarks and their insights paved the way for numerous derived approaches; even today most state-of-the-art pedestrian detectors still rely on HOG features albeit in a more subtle manner (e.g. in combination with other features).

A well-known example is the work of Felzenszwalb et al. [44, 46]. As opposed to the rigid model introduced by Dalal and Triggs, they propose to enrich these

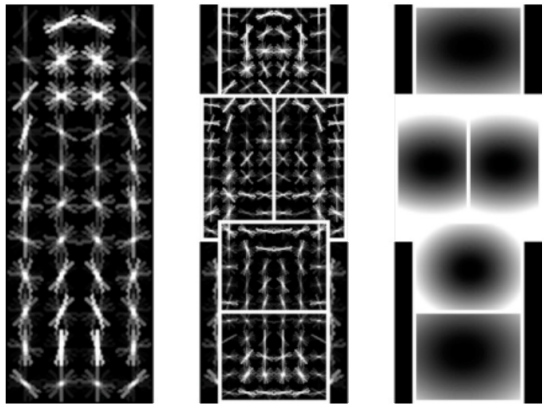


Figure 3.1: A deformable part model of a pedestrian. Left: Root HOG model. Middle: Different part filters representing the limbs and head of the pedestrian. Right: The deformation costs for each of the parts with respect to the root model.

rigid models using *parts* (representing e.g. the limbs or head of a pedestrian) to increase the detection accuracy, coined the *Deformable Part Models* (DPM) in 2008. Apart from the standard HOG model presented above (called the root model in the context of DPM), they include additional HOG filters which are calculated at a finer spatial resolution. These individual parts are allowed to *deform* slightly: their position with respect to the root model is not fixed but subjected to a specific deformation cost.

Figure 3.1 visualises a deformable part model of a pedestrian. The left image displays the root model, the middle image the different parts and the right image visualises the deformation cost of each part. Since slight deformations are allowed, their method performs much better for pedestrians due to the varying pose and viewpoint. Their method achieved state-of-the-art accuracy results on several object classes. The final detection score of a specific window that is evaluated is calculated as a combination of the root response and the part filter responses.

The response of these part filters are calculated at twice the resolution of the root filter response. The location of these specific parts are determined during the training phase using a *latent SVM*, hence this detector is often abbreviated as *LatSVM*. Such a complete deformable part model is called a *mixture model*, since it often consists of multiple components (representing the different viewpoints of an object). However, since these parts need to be detectable, a main disadvantage of these DPM detectors is that they require a

significant spatial resolution (small pedestrians are difficult to detect). To cope with this, Park et al. presented a method which combines a standard rigid HOG template for low-resolution with these DPM models for high resolutions (called *MultiResC*) [81]. A similar approach was presented by Yang et al. [124] where they employ resolution-aware transformations to map different resolutions to a common subspace.

However, including these parts evidently has a negative impact on the execution speed. In later work (2011) the authors tackled the inevitable increase in computational complexity by introducing a cascaded approach in which a fast rejection of negative detection windows is possible [43], much like in the work of Viola and Jones discussed above [117]. We effectively use this cascaded approach as a baseline detector in several chapters in this dissertation, since it offers several advantages that we can exploit as opposed to other pedestrian detectors. For more details we refer to chapter 4, subsection 4.2.2.

Furthermore, in 2012 an extension was proposed using grammar models to cope with partial occlusion [50, 52]. Several speed optimisations have been proposed. Dubout and Fleur proposed a framework that employs Fourier transformations to avoid the use of convolutions which often account for the main computational costs [37]. Pedersoli et al. present a coarse-to-fine approach where a low-resolution model is evaluated first, and more detailed models are only evaluated if needed [85]. In 2013, Girshick and Malik published a new and fast training methodology for DPM models [53]. Several other search-space optimisation techniques were proposed by Cho et al. [17] and Pedersoli et al. [84].

As opposed to enriching the model with parts (and thus increase the complexity of the model), a second approach exploits the use of other features (e.g. colour) besides the standard gradient features. Such work is presented by Dollár et al. in [32], which the authors coined *Integral Channel Features* (ICF). Here, they extend the rigid HOG model with several additional channels in which features are extracted. These image channels consist of a gradient magnitude channel, six gradient orientation channels and three LUV colour channels. Features are extracted as the sum of rectangular regions in one of these channels. These weak features are used in a depth-2 decision tree and learnt using AdaBoost. The exact distribution of these features is thus selected in an automated manner as opposed to the previous HOG based approach where this is done manually. Figure 3.2 displays an image patch with the resulting channels (top row) and the distribution of the features over the different channels. The orange rectangles, yellow arrows and red circle indicate the strong responses seen for respectively the arms, shoulders and head of a pedestrian.

This channel-based detection methodology lead to several derived and optimised pedestrian detectors that use a very similar approach. To speedup detection,

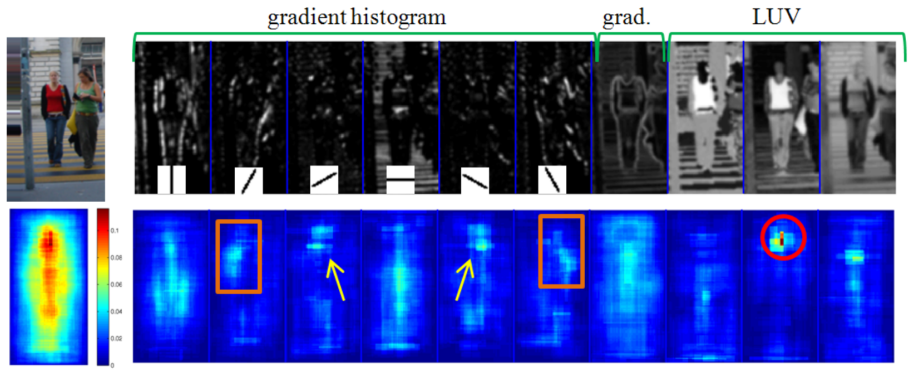


Figure 3.2: An example image patch with all calculated channels (top row) and the distribution of the selected rectangular features (bottom row). See text for details. Image from [32].

Dollár et al. proposed the *Fastest Pedestrian Detector in the West (FPDW)* in which only part of the feature pyramid was fully calculated, and intermediate layers were approximated from feature responses nearby [31]. In [6] Benenson et al. proposed their *Roerei* detector. They achieve state-of-the-art accuracy results by optimising each individual stage in the detection process. In the same work they propose the *SquaresChnFtrs* detector, where the variability of ICF (features are selected as random rectangles) is eliminated and only squares that fit inside the model window are used in the feature pool. In [30] an optimisation of the ICF detector – called *Aggregated Channel Features* (ACF) – was proposed where the channels are divided into blocks in which the pixels are summed. The features used are then simple single pixel lookups in the aggregated channels, which is extremely fast.

In [5] Benenson et al. presented work in which model rescaling as opposed to image scaling is performed: a model is trained for a number of layers, and intermediate models are approximated, thereby eliminating the need to construct a scale-space pyramid. They further exploited this concept on GPU hardware and – combined with their *stixel world approximation* [4, 8] – achieve pedestrian detection at 100 Hz in their publicly available *VeryFast* framework [5]. The authors claim that the *Roerei* detector is compatible with this framework, thus enabling excellent accuracy results with reasonable processing speeds. In [72] a methodology to cope with (partial) occlusions is proposed where different occlusion-specific classifiers are trained. Zhang et al. proposed a generalisation of the ICF detector coined the *Checkerboards* detector [127].

In [49] Girshick et al. present the use of *convolutional neural networks* (R-CNN) for object detection, achieving unprecedented state-of-the-art accuracy results. This methodology existed for a long time, but its applicability to image classification tasks was highlighted by the work of [65]. These convolutional neural networks are biologically inspired and are a special type of feed-forward artificial neural networks. They consist of multiple consecutive layers which transform the input data. Several different layer types exist, ranging from e.g. convolutional layers to pooling layers and fully-connected layers. A ConvNet *architecture* defines the exact structure of such a network. Examples are AlexNet [65], VGG [16, 98], ResNet [55] and GoogLeNet [102]. Interestingly, this methodology diverges from the traditional sliding window approach, and utilises region proposals as input for deep learning classifiers. These initial region proposals thus are crucial to obtain high detection accuracy [57, 58].

Although currently not real-time, these approaches are able to classify a large variety of classes simultaneously, making it ideal for large image database retrieval applications such as ImageNet [94]. These networks are able to learn most parameters themselves, only little human effort is needed. Although high accuracy is achieved, these convolutional neural networks have important disadvantages. A vast amount of image data is needed to train these networks and training is slow. Furthermore, the localisation of these neural networks is not optimal, i.e. they fail to return the exact position of the object of interest in the image. Extensive hardware is needed to achieve reasonable processing speeds (e.g. high-end GPUs or computer clusters) making them currently unsuitable for embedded implementations.

Often, integrations of standard pedestrian detectors and these R-CNNs are presented. For example, Girshick et al. present a hybrid approach combining DPMs with CNNs, called *DeepPyramid DPM* [51]. Hosang et al. present work where a pedestrian detector (*SquaresChnFtr*) is used to generate region proposals which are then classified with a convolutional neural network [59].

To summarise, in essence four main detection approaches currently exist: the Viola & Jones wavelet-based approach, the HOG-based rigid models, the deformable part models and the convolutional neural networks based approaches. Each of them has specific advantages. For several years, the DPM approaches remained among the top performing methods [34, 35]. However, the need of parts for pedestrian detection remains unclear [7]. Indeed, the more recent work on optimised rigid models – such as *Roerei* and *ACF* – in fact outperform the DPM based detectors. However, currently in general these CNN-based approaches achieve the best accuracy whereas the rigid detectors – e.g. ACF – have an excellent trade-off between accuracy and detection speed.

3.4 Comparing different methodologies

To allow for an accuracy comparison between different pedestrian detector methodologies, several renowned datasets are available. Examples are the INRIA [21], ETH [40], TUD-Brussels [121], KITTI [48] and the Caltech-USA [35] dataset. To facilitate these benchmark tests, Dollár et al. published an evaluation framework [34] in which accuracy results for many state-of-the-art pedestrian detectors are available (currently more than 50), allowing for a fair comparison.

The accuracy performance for these pedestrian detector is determined as follows. On a frame per frame basis each detection is evaluated. Every detection which is covered by a manually labelled annotation (called the *ground-truth*) is counted as a *true positive* (TP). A detection which could not be matched with a manual annotation is counted as a *false positive* (FP). Finally, a person which is not detected counts as a *false negative* (FN). Each detection is regarded as being covered by an annotation based on an overlap criterion. For this, often an intersection over union (IoU) of at least 50% is used [35, 42]. Given an annotation bounding box (A) and a detection bounding box (D), this IoU is thus given as:

$$IoU = \frac{area(A \cap D)}{area(A \cup D)} \tag{3.1}$$

Figure 3.3 displays an example frame from our dataset in which four VRUs are annotated (indicated with the dashed blue bounding boxes). In this frame two true positives (green bounding box), one false positive (red bounding box) and two false negative detections are found. Based on these measures we define the *precision* and *recall* as follows:

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

The precision thus indicates the percentage of correct detections (actual pedestrians) versus all detections. The recall indicates how many pedestrians the detector manages to find of all pedestrians available in the dataset. As previously mentioned each detector returns a detection score which indicates the certainty of a detection: the higher this score, the more likely it is that this image position indeed contains a pedestrian. By varying a threshold over these detection scores a *precision-recall* curve is constructed. At a low (i.e. *sloppy*) threshold value more pedestrians will be found at the cost of an increase in false detections (higher recall, lower precision). For a high (i.e. *strict*) value of

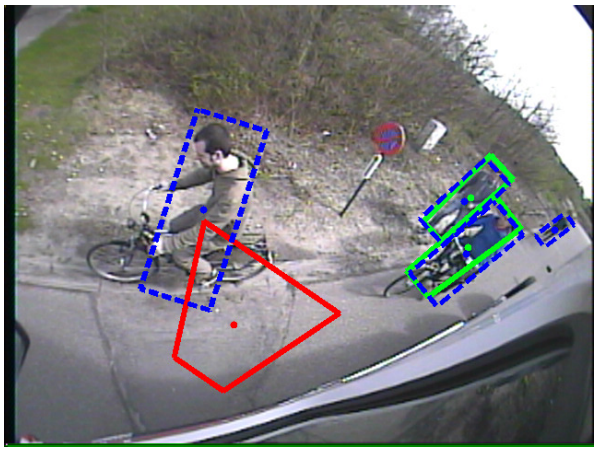


Figure 3.3: Example detections and annotations (dashed blue bounding boxes) on one of our dataset frames. Two true positives are found (green bounding boxes), one false positive is detected (red bounding box) and two VRUs are not detected (resulting in false negatives).

the threshold value the opposite is true: fewer pedestrians will be found, while the number of false detections decreases (lower recall, higher precision).

Figure 3.4 displays such a precision-recall curve for most pedestrian detectors which were discusses in the previous section (*Latv4-cc* represents the cascaded DPM [43], *ChnFtrs* represents ICF [32], *HOG* indicates the methodology of Dalal and Triggs [21] and *VJ* represents the Viola and Jones approach [117]). These results are achieved on the Caltech-USA dataset. This dataset is further subdivided based on several measures (e.g. level of occlusion, size of the labelled pedestrians and so on). Here we present the accuracy results on the *reasonable* setting, indicating that only pedestrians with a height of at least 50 pixels are included and of which at least 65% is visible. Note that depending on the size of the detection model for a specific pedestrian, sometimes image upscaling is needed to detect small pedestrians. The optimal point is found in the top right corner, when all pedestrians are retrieved with no false positive detections. The area under the curve (AUC) – displayed in the legend of figure 3.4 for each detector – thus effectively is an indication for the accuracy of a pedestrian detector and should be maximised.

Apart from the *precision-recall curves*, often *miss rate versus FPPI (false positives per image) curves* are used to compare the detection accuracy. We use the precision-recall accuracy measure in all remaining chapters throughout this dissertation, except for chapter 4, section 4.3. The miss rate (MR) is defined as:

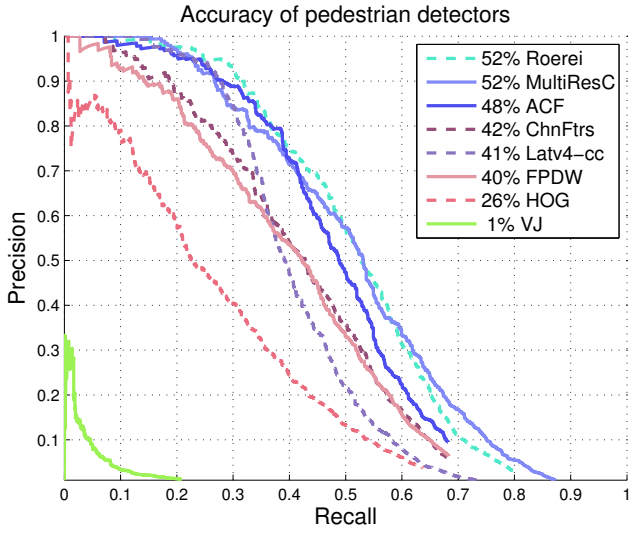


Figure 3.4: A precision-recall curve indicating the accuracy of different well-known pedestrian detectors on the Caltech-USA dataset [35].

$$Missrate = \frac{FN}{TP + FN} \tag{3.4}$$

The miss rate thus indicates the percentage of annotations that are not detected, whereas the FPPI indicates the average number of FPs per image frame. Evidently, in this case both measures should be minimised. We refer to figure 4.16 on page 69 for an example of such a MR versus FPPI curve.

3.5 Conclusion

In this chapter we gave an overview of the evolution and the state-of-the-art concerning pedestrian detection algorithms. Four main detection methodologies exist: wavelet-based approaches, the DPM models, rigid models and the more recent convolutional neural networks. We explained how these different detection methodologies are compared in a fair manner, and indicated their relative accuracy. This information serves as a background for the remainder of this dissertation.

In the next chapter we discuss a first step towards a complete vision-based VRU detection system for the blind spot zone: we propose a highly accurate and fast pedestrian tracking system for these specific blind spot images. For this, we first develop a basic tracking-by-detection approach for traditional backward-looking images recorded from a standard car. Next, we present a methodology that is able to cope with the large viewpoint and lens distortion induced by these blind spot cameras. Furthermore we present a highly optimised hybrid CPU/GPU implementation of this approach. Additionally, we developed a demonstrator which runs in real-time on embedded hardware.

Chapter 4

Pedestrian detection in the blind spot zone of trucks

In this chapter we propose our initial detection scheme that enables the detection of pedestrians in the blind spot zone of a truck. This chapter is subdivided into three main parts.

In section 4.1 we present an initial approach where we start from a standard backward-looking camera viewpoint and introduce an accurate and fast pedestrian *tracking-by-detection* framework for these images. The content of this section was published at the MVA 2011 conference [112] and the ATINER 2011 conference [105].

In section 4.2 we introduce an approach that enables the efficient detection of pedestrians in real blind spot images. Existing pedestrian detectors fail to achieve good detection results on these challenging images. We propose our *warping window approach* to cope with this specific viewpoint and high lens distortion. To develop and validate this approach, we acquired a challenging blind spot camera dataset with a real truck and genuine blind spot camera. For this, we simulated common dangerous blind spot scenarios. Our approach achieves excellent accuracy results. We presented this work at the ICINCO 2012 conference [113]. An extended version of this work was published as a book chapter in LNEE 2014 [114].

Finally, in section 4.3 we propose an efficient multi-threaded hybrid CPU/GPU implementation of our approach which achieves pedestrian detection at 500 detections per second. This work was co-authored with F. De Smedt

and published at the CVPR EVW workshop in 2013 [27]. Based on this implementation we developed a demonstrator of our blind spot detection system and show that even on an embedded platform real-time processing is achieved.

In this chapter we present an initial solution, i.e. the efficient detection of pedestrians. Next chapters will improve upon this chapter's initial solution: we will present several methodologies to further increase the detection accuracy of the approach presented here, and extend this approach towards multiclass detection.

4.1 An initial tracking-by-detection framework

4.1.1 Introduction

In this section we present our initial approach towards an accurate and fast pedestrian detection and tracking framework targeting the specific blind spot images. We first aim to develop a stable baseline tracking-by-detection framework. For this, in this section we employ standard backward-looking camera images, taken from a moving vehicle. This avoids the additional challenges induced by the blind spot camera (e.g. the viewpoint and lens distortion) while still maintaining the challenges involving the detection of moving objects from a moving camera. In section 4.2 we then propose our warping window approach that enables the detection of pedestrians in these genuine blind spot camera images, and discuss the integration of this approach in the tracking-by-detection framework presented here.

Most existing pedestrian tracking algorithms exploit the fact that the camera is often fixed, and rely on background subtraction (e.g. [96, 118]). Evidently, this is not feasible for our application. An existing approach for moving cameras is to exploit disparity characteristics (e.g. [47]). When using a monocular approach, most pedestrian trackers on moving vehicles use a forward-looking camera [40, 86]. We differ from these trackers: our final goal is a monocular multi-pedestrian tracking system with field of view aimed sidewards, towards the blind spot of the vehicle, at real-time performance. This field of view results in motion blur and large distortion.

The information from the appearance-based detector is used in combination with motion-based estimations to efficiently reduce the search space for the appearance-based detector in consecutive frames. We implemented and evaluated two different tracking approaches. The first pedestrian tracking algorithm is based on a tracking-by-detection Kalman framework. Here, after pedestrians are initially detected, their next positions in consecutive frames

are estimated using a specific motion model. Based on these estimated next positions, new appearance-based detections are only performed in limited regions.

The second tracking algorithm uses a multi-hypothesis approach: the probabilistic outcome of the detector (i.e. the detection score) is integrated in a particle framework. In consecutive frames, such detection score is calculated for each particle. Based on these scores the particles are reweighted. The technical details of both tracking frameworks are discussed in subsection 4.1.2. We performed several experiments to validate the usability of these two approaches with respect to both accuracy and detection speed. These results are discussed in subsection 4.1.3. In subsection 4.1.4 we present our conclusion regarding these tracking-by-detection frameworks.

4.1.2 Tracking-by-detection approach

Since we aim to achieve both high precision and recall we start from a highly accurate pedestrian detection technique. In essence, the tracking framework proposed here is independent of the pedestrian detector that is being used. Thus, all detectors discussed in chapter 3 are integratable in this tracking scheme. In this section we opted to use the cascaded deformable part-based model from Felzenszwalb et al. [43]. The main disadvantage of this detector as opposed to the rigid detection models is its computational complexity. Since one needs to search for pedestrians at each scale and position for both the root filter and the different parts, detection is indeed a time-consuming step. However, using the tracking frameworks discussed below we are able to significantly reduce the search-space, and thus speedup the actual detection time.

Kalman filter

Our first tracking algorithm employs a Kalman filter to include temporal information [63]. We define one or more initial search regions in the image where one expects that pedestrians enter the frame. In these initial search regions we evaluate the pedestrian detector mentioned above. If a pedestrian is found, the centroid of this detection is calculated, and a Kalman filter motion model is instantiated. A Kalman filter is an iterative process consisting of two distinctive steps: a *prediction step* and a *correction* step. A state vector x_k is stored for each detection. Based on a specific motion model (represented using a *state transition matrix* A) the next state is predicted. Then, in the correction step the estimated state vector is updated based on the *observation*. The difference between the estimated and observed state serves as an error

measure. In our framework we use a linear Kalman filter to estimate the next position of the pedestrian, based on a constant velocity model. Our experiments showed that this assumption holds and suffices for a robust detection. We use the position and velocity as our state estimates: $x_k = \begin{bmatrix} x & y & v_x & v_y \end{bmatrix}^T$. The Kalman filter is implemented with the following time update equation $\hat{x}_k^- = A\hat{x}_{k-1}$. Note that $\hat{x}_k^-$ refers to the *a priori* state estimate at timestep k, while $\hat{x}_k$ refers to the *a posteriori* state estimate at timestep k. We use a constant velocity motion model, and can only observe the position. The transition matrix A then becomes:

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Based on the *a priori* estimate $\hat{x}_k^-$ and the observation z_k the correction is performed as follows:

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \tag{4.2}$$

Here, K represents the *Kalman gain* (which is recalculated on each iteration) and H relates the effective state x_k to the measurement z_k. For more information we refer to [120]. Around the estimated new pedestrian centroid a circular region is constructed with a radius based on the position in the image. Detections which are closer to the horizon are given a smaller radius, since they are further away from the camera. This circular region is used to look for a new matching centroid in the next frame. A new search region is calculated around the estimated new centroid, of which the size is based on the extension of the previous bounding box area. For the consecutive frames we only look for pedestrians in the estimated search location, thereby reducing processing time and increasing the processing speed. Overlapping bounding boxes are combined into a single search space. The use of this search space also eliminates false detections, which could otherwise be found in the image where no pedestrians are possible. Figure 4.1 displays the initial and estimated next search space, together with the circular region in which a new centroid is expected.

After the search space is constructed we use our appearance-based pedestrian detector only on these parts of the image. For each pedestrian that is being tracked, we evaluate if a centroid of a new detection is found in the estimated circular region. If this is the case the existing tracker is matched with this detection. If multiple detections are found, the nearest one (based on the Euclidean distance) is chosen as a match. The Kalman filter is updated with this new information, a new bounding box is calculated based on a weighted average between the previous bounding box size and the current bounding box

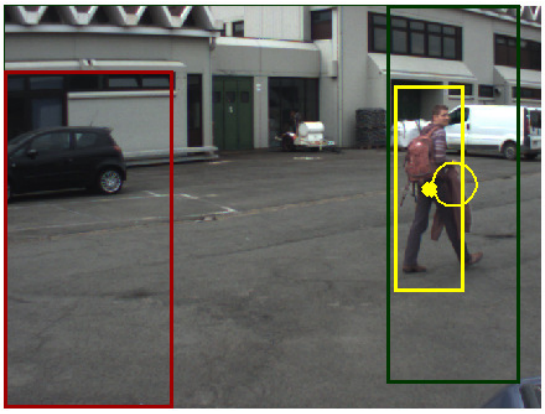


Figure 4.1: Example detection with estimated circular region (yellow), estimated next search space (green) and the initial search space (red).

size, and a new estimated position and circular region are calculated. If for a previous tracker no match is found, we update the Kalman filter based on our prediction. If this happens for multiple frames in a row, the track is discarded.

Evidently, when a detection is found where no previous tracker was available, a new track is initialised. Only pedestrians which can be tracked over multiple consecutive frames are shown as detected. We impose a number of application-specific constraints to improve the performance of our tracker: firstly, we reject detections of which the estimate is too far away from the centroid. Secondly, detections above the horizon and detections of which the size of the bounding box is inconsistent with the scale in the image are discarded. The latter is determined using the ratio of the bounding box size and the radius of the circular region.

Particle filter

The Kalman filter discussed above is only able to keep track of one specific hypothesis. This is a severe disadvantage, especially in the case of challenging environments where a detection is easily lost due to e.g. low contrast with the background (and thus low HOG features). Since the aforementioned deformable part detector returns a detection score which defines a similarity score between the image patch and the pedestrian model, we can exploit this score in a probabilistic framework. This allows for multi-hypothesis tracking, thereby increasing the accuracy. For this, we used a particle filter [3] to implement this probabilistic framework. As opposed to the Kalman filter implementation

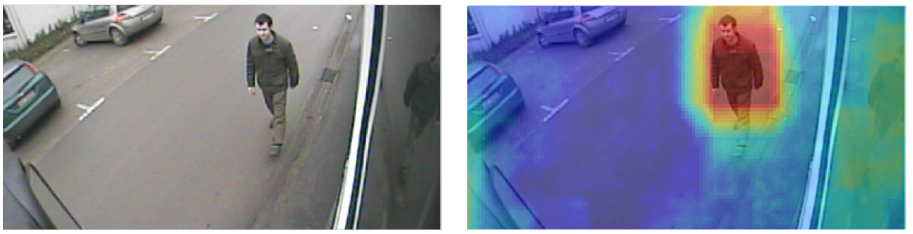


Figure 4.2: The likelihood of the presence of a pedestrian at each position in the image.

discussed above, these particle filters also allow for non-linear and non-Gaussian tracking. Furthermore, they are less prone for occlusions. Indeed, for the Kalman filter discussed above a fixed threshold for pedestrian detection is used. If – e.g. due to occlusion – the detection score for that pedestrian is below this fixed threshold, this detection is not retrieved. In this case, the Kalman filter relies on its prediction. If this happens for multiple frames in a row the track is lost. In the particle filter framework we employ a very low detection threshold to retrieve a detection score for each particle (as will be discussed below). As such, even occluded pedestrians (with a low detection score) are tracked. Particle filters approximate the probability density function $p(x_{0:t}|z_{1:t})$ of the state vector using discrete samples $\{x^i_{0:t}, i = 0, ..., N_s\}$ with associated weights $\{w^i_t, i = 1, ..., N_s\}$ up to time t as:

$$p(x_{0:t}|z_{1:t}) \approx \sum_{i=1}^{N_s} w^i_t \delta(x_{0:t} - x^i_{0:t}) \tag{4.3}$$

With normalised weights ($\sum_i w^i_t = 1$). Each sample, called a particle, is propagated using a specific motion model. When a new detection is found, the discrete sample set is updated based on the observation and resampled. Each particle thus correlates with a specific hypothetical state. The weight of each particle indicates how important the particle is. As motion model we used the same model as used for the Kalman filter implementation.

To reweight the sample set, for each particle we calculate the likelihood of the presence of a person at that location $p(z_t|x^i_t)$, using the deformable part model. The output of the detection model (i.e. the detection score) is used as new weight for the particles. As mentioned, to always retrieve a detection score for each particle, we employ a very low detection threshold. A state estimate is obtained as either the particle with the highest weight or (and more preferred) by calculating the weighted mean state based on all particles. As an example, figure 4.2 shows such a complete calculated likelihood heat map. Here, for

Figure 4.3: Example frames from our dataset. This dataset was recorded from a normal car with backward-looking camera at about eye-level.

each position in the image we evaluated the entire deformable part model, and returned the detection score (i.e. likelihood of their being a pedestrian) at that specific position.

4.1.3 Experiments and results

To evaluate both tracking frameworks we recorded a dataset from a moving vehicle with a standard camera, which consists of about 2000 frames in which both pedestrians and bicyclists are visible. The camera was mounted at eye-level and aimed backward-looking towards the blind spot region. Videos were recorded with both pedestrians and bicyclists. Figure 4.3 displays three example frames from our dataset to get an indication of the specific camera position. The dataset is divided into 12 short sequences. After editing we maintained 1279 walking pedestrian video frames. All quantitative evaluations presented here are performed on this dataset.

The frame rate of the camera equals 15 frames per second (FPS) at a resolution of 640×480. We have implemented our frameworks in Matlab and partially (that is, time consuming parts) in C. The results below are computed on an Intel Xeon Quad Core with a clock speed of 3 GHz, in a sequential, single-threaded manner. When using the publicly available vanilla DPM implementation [45] on an entire image frame, the detection time equals on average 2.8 seconds, i.e. 0.36 FPS. The publicly available cascaded DPM implementation requires on average 640 ms per frame (1.56 FPS) on this machine. Figure 4.4 displays a qualitative tracking sequence for both tracking methodologies (top row: Kalman tracking, bottom row: Particle filter). The individual particles are indicated with blue dots, the green dot indicates the weighted mean state.

Table 4.1 displays the speed results of our tracking algorithms on the first dataset, averaged over all sequences. For the particle filter we used 10 particles in our evaluation. We notice that our Kalman tracking algorithm greatly

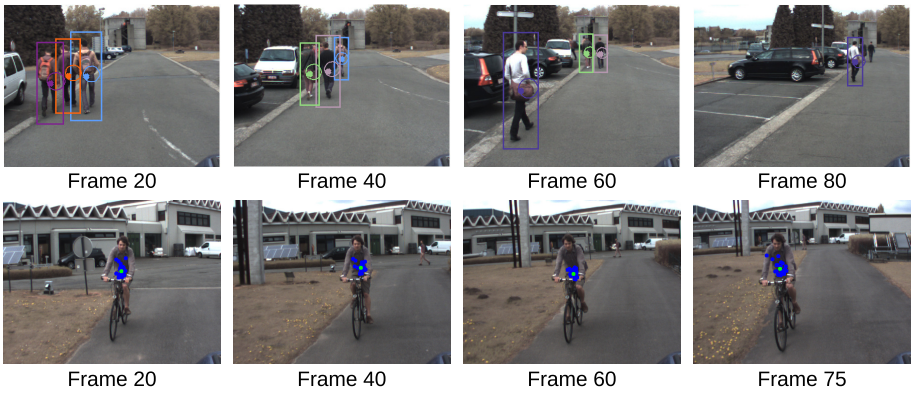


Figure 4.4: Qualitative tracking sequence for both tracking methodologies. Top row: Kalman filter. Bottom row: Particle filter.

reduces the computation time as opposed to the original vanilla implementation because of the reduced search space. The particle filter implementation has the advantage that multiple hypothesis are tracked (one for each particle), and that both non-linear systems and non-Gaussian distributions can be modelled. A disadvantage is that for each particle one needs to recalculate its weight. In our implementation we used the cascaded deformable part model to recalculate each weight, to increase invariance to object variations. This however comes at the cost of high computational complexity, which leads to low frame rates, as seen in table 4.1. To maintain reasonable computation times only simple weight recalculations can be used, such as for example colour histograms [25, 76].

Due to the high computational complexity of the particle filter as proposed here, this implementation is unsuited for real-time applications. Therefore, further accuracy experiments were only performed for the Kalman filter. Table 4.2 displays these accuracy results. Since these were only initial experiments on a preliminary dataset, the evaluation of the detection accuracy was performed for a single score threshold and only three sequences were manually labelled. The remaining sequences were only used for a qualitative analysis. Both a high precision and recall rate are achieved.

4.1.4 Conclusion

In this section we presented two tracking-by-detection approaches that we developed for a standard backward-looking dataset. These tracking frameworks were developed for high accuracy with real-time processing in mind. For this,

Table 4.1: Speed results for both our algorithm implementations.

Kalman filter approach	Average max. FPS	12.6 FPS
	Average min. FPS	2.8 FPS
	Average FPS	8.6 FPS
Particle filter approach	Average max. FPS	0.25 FPS
	Average min. FPS	0.11 FPS
	Average FPS	0.22 FPS

Table 4.2: Accuracy results for Kalman filter approach.

	avg. FPS	**precision**	**recall**
seq. 1	9.57	0.76	0.92
seq. 2	8.97	0.95	0.93
seq. 3	9.47	0.95	0.8
average	9.34	0.89	0.88

these frameworks rely on a highly accurate but computationally intensive pedestrian detector. Both tracking approaches aim to reduce the search space of this pedestrian detector – using temporal information – and as such render the pedestrian detector useful for real-time applications.

Our experiments indicated that the computational complexity of the particle-based tracking framework as presented here – where each weight is calculated as a detector response – was unsuited for real-time applications. Using the Kalman-based tracking approach reasonable processing speeds at excellent accuracy results were obtained. Therefore, this tracking methodology will serve as baseline tracking framework for the remainder of this dissertation.

In the next section we now present an approach that enables the efficient detection of pedestrians in the challenging blind spot camera images. As will be discussed, to further extend the accuracy of this detection approach, we integrate it in the baseline tracking framework presented here.

4.2 Detecting pedestrians in genuine blind spot camera images

4.2.1 Introduction

In this section we present our approach that enables the detection of pedestrians in the challenging blind spot camera images. Existing pedestrians detectors fail to reliably detect pedestrians in these images due to the specific viewpoint and large lens distortion. To cope with the challenges introduced when using a genuine blind spot camera we present our *warping window approach*. This approach enables accurate and fast pedestrian detection. We integrate this approach in the baseline tracking framework discussed above, and thus present a real-time robust multi-pedestrian detector and tracker for real blind spot camera images which achieves high accuracy.

Our algorithm achieves real-time performance while maintaining high accuracy. To evaluate our algorithm we recorded several pedestrian datasets with a real blind spot camera mounted on a real truck, consisting of realistic simulated dangerous blind spot situations.

Note that in this chapter we present an initial solution and only target the detection of pedestrians in the blind spot zone. Throughout the next chapters we will present additional methodologies which further increase the accuracy of this approach and enable multiclass detection.

As opposed to the classically used sliding window approach, our algorithm is based on a *warping window approach*. As mentioned, this warping window approach aims to overcome the challenges induced by the specific viewing angle of a real blind spot camera mounted on a real truck, and the distortion that this camera introduces. An example frame of our blind spot camera setup is displayed in figure 4.5. We refer to chapter 2, section 2.7 for more details on our specific camera setup. One clearly sees that standard pedestrian detectors, even if they were fast enough, cannot be used on these images because they are developed for pedestrians that appear upright in the image. Using our framework we manage to robustly detect and track the pedestrians while maintaining excellent speed performance.

This is briefly done as follows. Using our warping window method, we can warp the regions of interest in the image and use a standard pedestrian detector at only one specific scale, which is very fast. We then integrate this approach in the tracking-by-detection framework discussed above, and further speedup the algorithm using temporal information to reduce the search space. To meet the strict accuracy demands, we use a pedestrian detector which has very

Figure 4.5: Example frame of our blind spot camera setup.

good accuracy at the cost of high computation time when it is used as is. Using our framework this detector still achieves high accuracy but at real-time performance (on our dataset we achieve an average frame rate of 10 FPS). Since to our knowledge no truck blind spot camera datasets are available in the literature, we recorded our own real-life datasets in which we simulated different dangerous blind spot scenarios using a real truck. These images are used to evaluate our algorithm regarding both speed and accuracy.

The remainder of this section is structured as follows. In subsection 4.2.2 we present our warping window approach, discuss which pedestrian detector is most suited and present the integration of our approach in the tracking-by-detection framework. Next, we present extensive experiments with respect to both accuracy and speed in subsection 4.2.3. Finally, this section is concluded in subsection 4.2.4.

4.2.2 Warping window approach

Our warping window algorithm is mainly based on the following observation. Looking at the blind spot camera example frame in figure 4.5 one clearly notices that, due to the specific position of the blind spot camera and the wide angle lens, pedestrians appear rotated and scaled. The crux of the matter is that the amount of rotation and scaling is only dependent on the position in the image because the camera is positioned on a constant relative position with respect to the (moving) ground plane. Thus, each pixel coordinate $\mathbf{x} = [x, y]$

represents a specific scale and pedestrian rotation. If at each pixel coordinate the corresponding rotation and scale is known, one can dramatically speedup pedestrian detection. Instead of a classic full scale-space search we can warp the region of interest, which is extracted based on the scale at that pixel coordinate, to upright pedestrians on one standard scale. This way we can use a standard pedestrian detector at only one scale, which is very fast.

Besides our application, this approach can easily be generalised to other applications where such wide-angle distortion and/or non-standard camera viewpoints occurs (e.g. surveillance applications), as we will illustrate in chapter 5. To get the rotation and scale for each pixel coordinate a one-time calibration step is needed. To enable robust tracking we integrate this warping window approach into our tracking-by-detection framework presented in subsection 4.1.2. We use temporal information to predict the next pedestrian positions, eliminating the need for a full search over the entire image. We now describe each part of this approach. First, our warping window approach is described in detail. We then give a quantitative motivation for our pedestrian detector choice and the size of our standard scale. Finally, we explain how we integrate our warping window approach into a robust tracking framework, and thus describe how our complete framework works.

Warping Window Approach

The warping window approach is visualised in figure 4.6. Given input images as in figure 4.5, the pedestrians appear rotated and scaled at different positions in the image. If we assume that we have a flat ground plane, we know that the rotation and the scale of these pedestrians only depend on their position in the image. Thus if the scale s and rotation θ are known for each position in the image (visualised in the figure using the 2D *lookup functions* or LUF heat map plots), we can warp the pedestrian ROIs (I) into upright pedestrians at a standard scale (I_{warp}), using $I_{warp} = TI$, with transformation matrix T:

$$T = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

A one-scale detector is used to detect the pedestrians, and the output coordinates of the bounding boxes are retransformed into input image coordinates. These coordinates are then fed into our tracking framework, to determine the next pedestrian ROIs. To determine the scale and rotation for each pixel coordinate, a one-time calibration step is needed. To achieve this, we manually labelled about 100 pedestrians in the calibration images homogeneously spread over

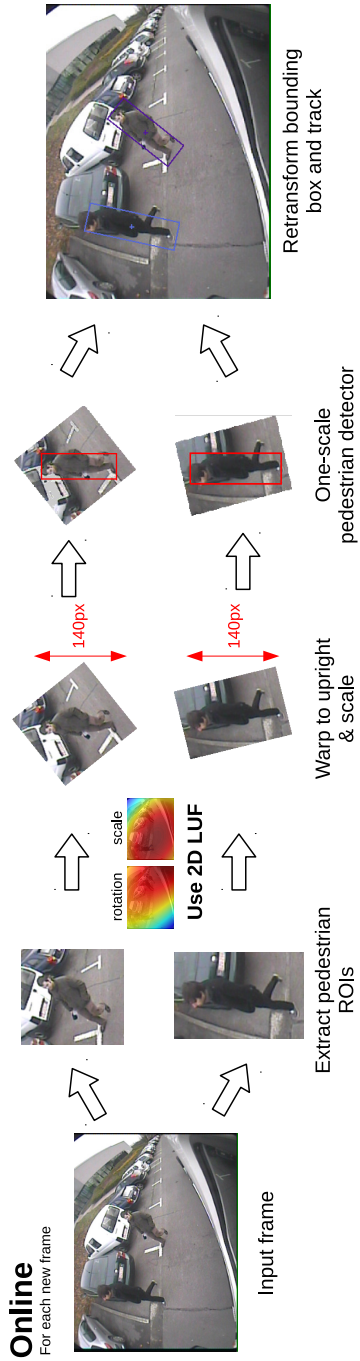


Figure 4.6: Our warping window approach. If the scale and rotation are known, we can warp the ROIs and use a standard pedestrian detector at only one scale. Detecting pedestrians is composed of four steps: extract the pedestrian ROI, calculate the scale and rotation for that ROI, retransform to an upright pedestrian with a standard height of 140 pixels and use a pedestrian detector at only one scale.

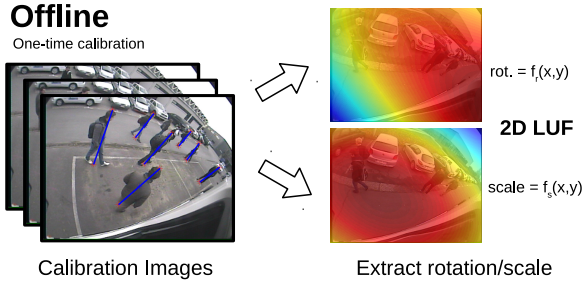


Figure 4.7: A one-time calibration step is needed.

the total image region. Each pedestrian yields scale and rotation data at that position. Next we fitted a two-dimensional second order polynomial function through the data points: $rotation = f_r(x, y)$ and $scale = f_s(x, y)$ where:

$$f_i(x, y) = A + Bx + Cy + Dx^2 + Exy + Fy^2 \tag{4.5}$$

Both functions are visualised as the two heat maps in figure 4.7. These two functions effectively represent a 2D lookup function, i.e. for each pixel coordinate they give the rotation and scale at that pixel position. If the camera position is adjusted, we need to perform a recalibration. However, due to the robust camera mounting on the truck this occurs only rarely.

Thus detecting pedestrians is composed of four steps: extract the pedestrian ROI, calculate the scale and rotation for that ROI, retransform to an upright pedestrian with a standard height of 140 pixels and use a pedestrian detector at only one scale. The choice for this number is motivated below.

Pedestrian detector

Since we only need to detect pedestrians at a standard scale (140 pixels), our approach allows the use of a detector with high accuracy which would otherwise be too computationally expensive. Given the extensive comparative results that we discussed in chapter 3, section 3.3, two pedestrian detector methodologies are applicable in our framework. We exclude the R-CNN based detectors (since currently real-time operation is only achieved on extensive hardware) and the Viola & Jones based approach due to the low detection accuracy. Both the deformable part-based detectors (DPM) and the rigid detectors achieve high accuracy. Currently, the accuracy of the part-based models is slightly lower, and the computation time is higher due to the scale-space pyramid construction.

One would thus assume that a rigid detector is best suited to be used in our framework. However, since we only perform pedestrian detection at a single scale, no such scale-space pyramid needs to be constructed in our application. These rigid detectors achieve the best accuracy on standard datasets. Furthermore, due to the part-based approach, the DPM detection model is much more flexible, making this detector invariant to slight deviations in height between the pedestrians that need to be detected, and the actual pedestrian model. The rigid detection models are much more sensitive for this (evidently due to the rigidness of their detection model). Since in our image patches slight differences between the actual and estimated pedestrian height exist (due to small calibration errors and the inherent height differences between pedestrians), more search scales would be needed to obtain a good accuracy with the rigid detection approach. We prove these assumptions with extensive experiments in chapter 7, subsection 7.1.3.

Our choice thus goes to the deformable part-based detector introduced by Felzenszwalb et al. [44, 46]. As a reminder, let us now briefly summarise how this pedestrian detector works if used out-of-the-box. The object that has to be detected is described using a HOG model. The model consists of a root filter, representing the pedestrian appearance, and a number of smaller part filters, representing the head and limbs of the pedestrian (see figure 3.1 in chapter 3 on page 33). The position of each of the parts are latent variables, which are optimised during the detection. A first step is the construction of a scale-space pyramid from the original image. This is done by repeated smoothing and subsampling. For each entry of this pyramid, a feature map is computed, which is built using a variation of the HOG features presented by Dalal and Triggs [21]. For a specific scale one computes the response of the root filter and the feature map, combined with the response of the part filters and the feature map at twice the resolution at that scale. The transformed responses of the part filters are then combined with the response of the root filter to calculate a final detection score.

As a reference, if used out of the box on our images (640×480 resolution) this detector needs an average of 2.8 seconds per frame (evaluated on a Intel Xeon Quad Core running at 3 GHz, all implementations are CPU-based only). If we reduce the number of scales to only contain those needed in our application, detection time decreases to about 850 ms. Later, Felzenszwalb et al. presented a cascaded version [43]. There, using a weak hypothesis first, a fast rejection is possible while maintaining accuracy. Using this detector, again out of the box and only on the scales needed in our application, the detection time on our images equals 640 ms.

We altered both the default and the cascaded part-based pedestrian detector to a one-scale detector. In figure 4.8 the average calculation times of the four

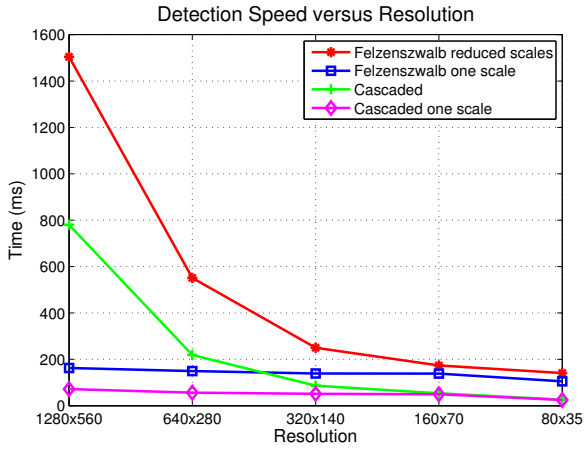


Figure 4.8: The calculation time per patch for the different pedestrian detector implementations.

different implementations, namely the part-based model with reduced scales (further referenced as *Felzenszwalb reduced scales*), our one-scale implementation of this detector (referenced as *Felzenszwalb one scale*), the cascaded version and our one-scale implementation of the cascaded version. Needless to say, the detection time strongly depends on the image resolution. To generate figure 4.8, we used a high resolution pedestrian image and cropped the image to a patch only containing the pedestrian. This patch was then subsampled to the indicated resolutions. Calculation times are averaged over ten runs. Note that to obtain a fair comparison we deliberately did not cache any data. For example, the pedestrian model is completely reloaded into memory on each run. We can clearly see that decreasing the resolution drastically reduces the calculation time for both the standard *Felzenszwalb* and the cascaded implementations. The calculation time of our one-scale implementations does decrease with resolution, but not nearly that fast. Since only one scale is looked at, a double gain in speed is realised. The scale-space pyramid does not need to be constructed, and features only need to be calculated and evaluated at one scale. In our warping window framework we use this cascaded one-scale detector.

Reducing the resolution implies that the accuracy drops significantly. Therefore we needed to determine the optimal trade-off point between the detection accuracy and the resolution to which we warp our pedestrian images. To determine that optimal resolution we extracted about 1000 pedestrians from our dataset, rescaled them to fixed resolutions and determined the accuracy of our one-scale cascaded detector for each resolution. These results are displayed

Figure 4.9: The accuracy of our one-scale cascade detector implementation in function of the pedestrian resolution.

in figure 4.9. Here, we define the accuracy as the *true positive rate* ($TPR = \frac{TP}{TP+FN}$). At higher pedestrian resolutions the accuracy remains almost constant at around 94%. When decreasing the pedestrian resolution the accuracy starts to drop at approximately 135 pixels. Based on these observations we chose to rescale our pedestrians to a constant standard height of 140 pixels in our warping window approach. This results in an average calculation time of 45 ms per patch when using the one-scale cascaded detector. If the model does not need to be reloaded on each run, calculation time further decreases to about 12 ms per patch.

Integration in the tracking-by-detection framework

Our complete blind spot pedestrian tracking-by-detection algorithm now works as follows. We integrate our warping window approach into a reliable tracking-by-detection framework, similar as discussed in subsection 4.1.2. This is done as follows. At positions where pedestrians are expected to enter the blind spot zone in the frame, initial (i.e. standard) search coordinates are defined (see figure 4.10).

Our warping window approach is used to detect pedestrians at these search locations. If a pedestrian is detected, tracking starts. For this, we employ the exact same Kalman tracker (constant velocity model, and the same state matrix A). Our experiments show that this assumption holds and suffices for

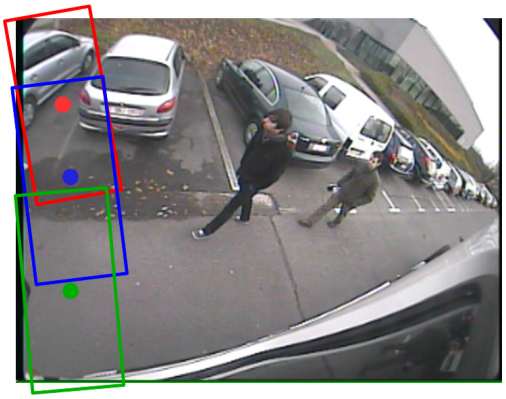


Figure 4.10: Example of three initial search coordinates, together with the initial search regions that they define.

a robust detection. Using the motion model we predict the position (that is, the centre of mass) of the pedestrian in the next frame. In subsequent frames we use the estimated centroids and the standard search coordinates as inputs for our warping window approach. For each estimated centroid and standard search coordinate our warping window approach warps this ROI to an upright pedestrian at a fixed scale and performs pedestrian detection.

For each pedestrian that is being tracked, our algorithm verifies if a new detection is found. This is evaluated by constructing a circular region around the estimated coordinate with a radius based on the scale at that coordinate, determined from the 2D scale LUF. If a new detection is found in this region, the Kalman filter is updated and the new position is predicted. If multiple detections are found, we associate the closest based on the Euclidean distance.

The bounding box coordinates of tracked instances are averaged over two frames to assure smooth transitions between frames. If for tracked pedestrians no new detection is found, the Kalman filter is updated based on the estimated position. In this case we apply a dynamic score strategy, and lower the detection threshold for that instance (within certain boundaries). This ensures that pedestrians which are difficult to detect (e.g. partially occluded or a temporarily low HOG response) can still be tracked. If no detection is found for multiple frames in a row, the tracker is discarded. Evidently, if a detection is found with no previous tracked instance, tracking starts from there on. Indeed, apart from the initial search regions, a new track can be started as follows. As mentioned above, for each estimated next position of a pedestrian, we extract an image patch at that predicted location. Next, pedestrian detection is performed on this image

Figure 4.11: Example output of our tracking algorithm.

patch based on a sliding window methodology. Multiple pedestrians might be found in the same image patch (e.g. when due to occlusions the pedestrian was invisible for multiple previous frames). After NMS, each remaining detection is then matched with the currently running tracks. If for a specific detection no match is found, a new track is started from there on. This tracking-by-detection approach eliminates the need for a full frame detection, thus limiting processing time. Figure 4.11 shows the output of our warping window tracking algorithm on one video sequence.

4.2.3 Experiments and results

Due to the specific viewing angle of the blind spot camera no image datasets are available in the literature. Therefore we constructed such a dataset, consisting of several simulated dangerous blind spot scenarios. This was done using our real blind spot camera, mounted on a real truck. As mentioned, for this we used a commercial blind spot camera (Orlaco CCC115°), which outputs 640×480 images at 15 frames per second. It has a wide-angle lens with a viewing angle of 115 degrees. For the exact position of the blind spot camera on our truck we refer to figure 2.10 in chapter 2. We recorded five different scenarios. At each scenario the truck driver makes a right turn, and the pedestrians react differently. For example, in some of the scenarios the truck driver takes a right turn while stopping to let the pedestrians cross the street, while in other scenarios the pedestrians stand still at the very last moment while the truck continues his turn.

These simulations resulted in a dataset of about 11000 frames. Again, our evaluation hardware consists of an Intel Xeon Quad Core, which runs at a clock speed of 3 GHz. All implementations are CPU-based, we do not use GPU implementations. The algorithm is mainly implemented using Matlab, while part of the pedestrian detector is implemented in standard C-code. The image warping is implemented in OpenCV, using *mexopencv* [123]. As mentioned in the previous section, as a reference, when used out of the box the Felzenszwalb

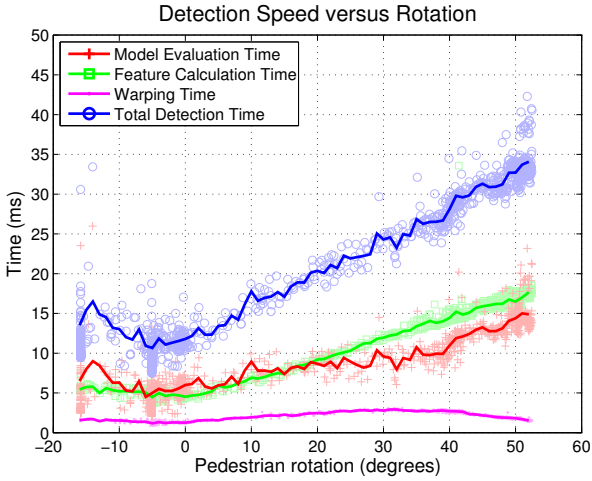


Figure 4.12: Speed analysis of our warping window approach. The blue line indicates the total calculation time per pedestrian, in function of the rotation.

pedestrian detector needs 2.8 seconds for a full scale-space detection over an entire frame. As our goal is to develop a real-time pedestrian tracker with high accuracy, we evaluated the algorithm with respect to both speed and accuracy.

Speed analysis

For each tracked pedestrian we need to perform a new detection in the consecutive frames. Thus if more pedestrians enter the frame, the total calculation time increases. Figure 4.12 displays the detection time per tracked pedestrian in function of the rotation. We split up the total detection time in three separate steps: first the image patch is warped in an upright fixed scale pedestrian image patch. Then our pedestrian detector calculates the HOG features. The last step consists of the actual model evaluation, in which the image is given a score based on the DPM model. The total detection time increases if the rotation angle increases. Warping the patch is computationally the least expensive operation. It only slightly depends on the rotational value, and maximally takes about 3 ms. The feature calculation and the model evaluation take almost an equal amount of time, and both increase with increasing rotation. This is due to the fact that the total image area increases with increasing rotation (see figure 4.13). If no rotation is needed, both feature calculation and model evaluation time take about 5 ms, resulting in a total detection time of 12 ms. In the worst-case scenario, occurring at a rotation of 52 degrees (the

Figure 4.13: Example pedestrian detector input images for different rotations.

Table 4.3: Speed Results as measured over our dataset.

	best-case	**average**	**worst-case**
FPS	50.8	10.1	7.8
# pedestrians	0	3.1	5

maximum rotation in our application), the detection time increases to 35 ms. Thus if e.g. two pedestrians are tracked, of which one at low rotation and one at high rotation, detection time for these pedestrians requires about 45 ms. Evidently, the fact that the detection time depends on the degree of rotation is only an implementation issue. After the image patch is extracted and rotated, only the central part of the patch contains useful information and thus the image could be cropped as such. In our implementations proposed in chapter 7 this warp step is optimised to avoid this rotation dependent calculation time. If two standard search regions are included at e.g. 15 ms each the total frame detection time equals 72 ms. In that case the algorithm achieves a frame rate of 14 frames per second. If multiple pedestrians are detected, detection speed decreases. Large groups of pedestrians are however easily noticed by the truck driver and therefore do not pose a real risk for accidents. Most blind spot accidents occur when only a few (mostly only one) pedestrian are in the blind spot zone. If only one pedestrian is tracked our algorithm achieves a frame rate of more than 20 frames per second. Table 4.3 shows the average, best-case and worst-case frame rate as evaluated over our dataset, and gives the number of pedestrians that were tracked while achieving these frame rates. To obtain these results, we used three initial search coordinates. Since in our dataset on average more than three pedestrians were visible per frame, the average calculation time given here is in fact an overestimation of the calculation time for a real scenario (since most blind spot accidents occur when only one or a few pedestrians are present in the blind spot zone).

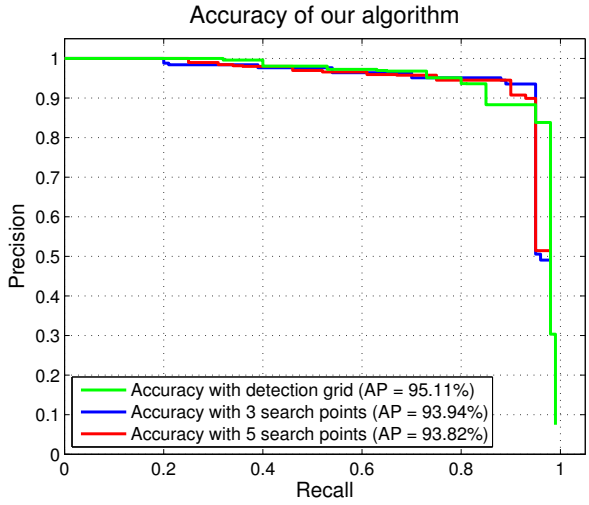


Figure 4.14: A precision-recall curve of our algorithm as evaluated over our dataset. Different colours indicate accuracy results when using a different number of default search regions.

Accuracy analysis

The accuracy of our detector is displayed in the precision-recall curve in figure 4.14. For each pedestrian that our algorithm detects, we look for the centroid of a labelled pedestrian in the circular region of the detection. If this is the case, the detection is counted as a *true positive*. If no labelled pedestrian is found, the detection is indicated as being a *false positive*. If a labelled pedestrian is not detected, this is indicated as being a *false negative*. Our test set consists of approximately 1200 frames in which about 3200 pedestrians are labelled, in very diverse poses and movements. We give the precision and recall for a different number of initial search points (3 search coordinates, 5 search coordinates and a grid consisting of 14 points homogeneously spread over the entire image region). As can be seen in the figure our algorithm achieves both high precision and recall rates. Increasing the number of initial search coordinates has only limited influence on the accuracy, indicating that our tracking algorithm efficiently manages to track all pedestrians when they enter the initial search regions. The average precision (AP) – calculated as the area under the curve (AUC) is given in the legend of figure 4.14. As seen, we achieve excellent accuracy results – around 94% when only using three initial search coordinates. At a recall rate of 95%, we still achieve a precision rate of 93.6%. This is due to the fact that using our warping window approach, the specific scale at each position is known.

Therefore false positives are minimised, while the pedestrian detection threshold can be set very sensitive. This way difficult to detect pedestrians can still be tracked. While very good, the accuracy is not perfect yet. Our warping window approach sometimes fails to track pedestrians due to low responses of the HOG features, induced because only a subtle intensity difference between the pedestrian and the background occasionally occurs. A possible solution for this is the inclusion of other features, e.g. motion information.

4.2.4 Conclusion

In this section we presented our warping window approach which allows for the tracking of multiple pedestrians in the blind spot camera images. We introduced this warping window approach to cope with the specific wide-angle distortion induced by the blind spot camera. Moreover, this methodology is easily applicable to other object detection applications in situations where such distortion occurs, e.g. caused by non-standard camera viewpoints or specific lenses as we will show in the next chapter. To evaluate our algorithms we recorded a representative real blind spot dataset. Experiments were performed evaluating both the speed and accuracy of our approach. Our algorithm achieves real-time performance while maintaining both high precision and recall.

Keep in mind that the approach presented here is only an initial solution towards the detection of VRUs in the blind spot zone. In the next chapters we further refine this approach to increase the detection accuracy, and allow for multiclass detection. Furthermore we present methodologies that enable a more deterministic calculation time and enable the detection of children. Finally, we elevate this approach into a final active alarm system and present accuracy and usability results as such.

In the next section we discuss an optimised implementation of our approach achieving 500 detections per second. Using this implementation we developed a real-time demonstrator of our detection framework.

4.3 A fast and efficient hybrid implementation

In this section we propose an efficient real-time implementation of our warping window approach discussed above. For this, we developed an implementation which integrates our warping window approach in a hybrid multi-threaded CPU and GPU implementation. This complete detection framework allows for real-time operation – even on an embedded system – as we will discuss below.

Additionally, we provide experimental results in which we illustrate the accuracy gain when using our warping window approach on a publicly available dataset (Caltech [34, 35]).

4.3.1 Introduction

As previously mentioned, a pedestrian detector which is at the same time fast and accurate would open up a wide variety of applications, including robotics, surveillance and automotive safety. These applications clearly benefit from the high robustness most recent algorithms can achieve. Indeed, over the past few years impressive accuracy improvements were obtained on challenging benchmark datasets, containing a wide variety of poses and appearances. Unfortunately, high accuracy often comes at the cost of high computation time, making these algorithms unfeasible in real-life applications. In this section, we overcome this problem through algorithmic optimisation and the exploitation of scene constraints. We present an efficient pipelined hybrid CPU/GPU pedestrian detector implementation.

While being fast on its own, we further improve the speedup using the integration of the warping window approach. As mentioned, this approach allows to limit the search space and thereby reduces both the false positive rate and the detection time while allowing arbitrary non-linear camera distortion and extreme viewing angles. We benchmark our hybrid pedestrian detector implementation both with respect to accuracy as to speed on the Caltech dataset, and show that despite the speedup, we achieve state-of-the-art accuracy. Moreover, we quantitatively demonstrate how the warping window approach further increases the accuracy on the Caltech dataset. Finally, we illustrate the full potential of this warping window approach, and achieve excellent accuracy results with very high processing speeds (500 detections per second). Such speeds are useful for e.g. crowded scenes or when implementing on embedded hardware with limited processing power.

The remainder of this section is structured as follows. We first describe our hybrid CPU/GPU pedestrian detector in subsection 4.3.2. In subsection 4.3.3 we discuss the quantitative accuracy improvement results obtained when using this warping window approach. We then combine both approaches in subsection 4.3.4, and show how our speedup is realised. In subsection 4.3.5 we discuss the real-time demonstrator that we developed, and give additional speed measurements of our approach on three different platforms, including one embedded board. Finally, we present our conclusions in subsection 4.3.6.

Apart from the demonstrator presented in subsection 4.3.5, we published the work presented in this section in [27]. This publication was co-authored

with F. De Smedt, who contributed the hybrid CPU/GPU implementation (subsection 4.3.2) in which the warping window approach is integrated. The obtained accuracy improvements (subsection 4.3.3) were contributed by K. Van Beeck. The final integration presented in subsection 4.3.4 was joint work.

4.3.2 Hybrid pedestrian detector

Speed improvement can be obtained in two ways: either optimise the execution or decrease the search space (or a combination of both). In this subsection we discuss the first, while in the next subsection (subsection 4.3.3) we target the latter. The accuracy and speed results discussed in this section therefore are measured without scene constraints, thus applying a full scale-space search. The object detection algorithm we start from is again based on the *cascaded DPM detector* proposed by Felzenszwalb et al. [43]. This detector achieves state-of-the-art accuracy results. Using this original part-based pedestrian detector implementation [45] as a baseline, we propose two new implementations.

In the first implementation we ported the calculation of the feature pyramid to the GPU, resulting in a significant speedup. Our second implementation consists of a multi-threaded hybrid CPU/GPU implementation, achieving a speedup of 12.7× over the original implementation. In subsection 4.3.4 we then integrate this last implementation with scene constraints and temporal information, and propose our final *WSPD* (*Warp Speed Pedestrian Detector*), which achieves pedestrian detection at 500 detections per second. For the remainder of this subsection we discuss our first implementation (feature pyramid on GPU), the multi-threaded hybrid CPU/GPU implementation and give evaluation results concerning both speed and accuracy.

GPU Feature pyramid implementation

The feature pyramid consists of multiple layers, each containing the features of a rescaled version of the source image. The publicly available DPM implementation (referred further as *LatSVMV4 (original)* - referring to the latent variables) does not suffice for real-time applications. Therefore as a first step towards a GPU implementation we reimplemented the algorithm to C++. Since GPU hardware benefits from the use of floats, our C++ reimplementation (further referred to as *WSPD-v0.1 (float)*) uses *float* variables (as opposed to the original *double* implementation). One could argue that this leads to a decrease in accuracy. We show however, that this is not the case. Based on this C++ implementation we implemented the feature pyramid calculation on GPU using CUDA. In essence CUDA is a C extension that allows the use of nVidia GPU

hardware as an execution device, enabling faster execution of algorithms that use data parallelism. In previous work De Smedt et al. implemented the feature pyramid in OpenCL [26], however the new implementation proposed here is faster due to the use of this hardware specific language. We further refer to this GPU implementation as *WSPD-v0.2 (GPU)*.

Multi-threaded hybrid implementation

Although the use of GPU hardware allows for a significant speedup, it does not fully exploit the capabilities of the hardware system, since the CPU is only active when the GPU is idle and vice versa. To further speedup the algorithm, and to circumvent this problem, we propose an implementation using a pipelined object detection scheme. In particular, we calculate both the feature pyramid of a frame (on GPU) while at the same time performing the model evaluation step for each layer in the feature pyramid from the previous frame (on CPU). This way the CPU and GPU are both active as a hybrid system, allowing an increased detection throughput. Besides running the feature pyramid and model evaluation in parallel on CPU and GPU, we can further increase the detection throughput by matching the execution time of each step in the pipeline. For example, the feature pyramid is almost twice as fast as the model evaluation step, so for each feature pyramid process we run two model evaluation processes, leading to a higher throughput. An even faster detection throughput is achieved when running multiple instances of the detection pipeline in parallel. Figure 4.15 shows a schematic overview of our implementation structure, which we further refer to as *WSPD-v0.3 (hybrid)*. This structure focuses on the use of eight threads (one per block). As shown, we use two detection pipelines in parallel. We will discuss each block of the schematic overview in more detail below.

Preprocessing. Here, all preprocessing (e.g. cropping, rescaling, rotating and search space pruning) is performed. The results are placed in the image queue for further processing.

Feature Pyramid. In this block the feature pyramid is calculated. The images are gathered from the image queue, while the feature pyramids are pushed into the pyramid queues. Since the calculation of the feature pyramids is executed on GPU, the data transfer to the GPU is also included here. In the schematic overview of figure 4.15, two instances of the feature pyramid are displayed, indicating that the feature pyramid of two frames is computed in parallel. Due to the modular approach of our innovative hybrid detecting scheme the implementation is easily hardware scalable. In our experiments each

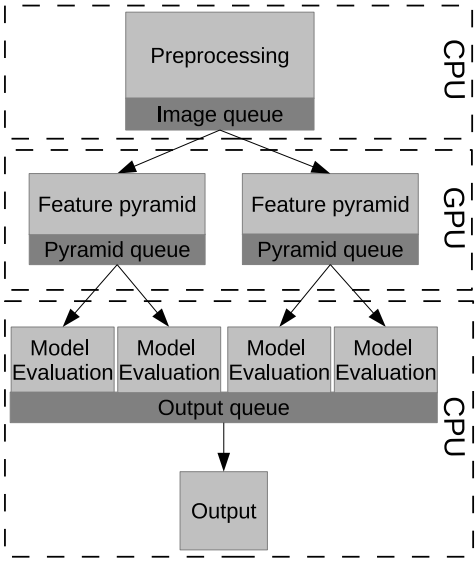


Figure 4.15: A schematic overview of the hybrid detector.

feature pyramid instance ran on a separate CUDA device (our GPU includes two such devices). Although it is possible to run multiple instances on the same CUDA device, this is evidently limited by the GPU resources.

Model Evaluation. For each calculated feature pyramid, we need to evaluate the pre-trained model on each layer in the pyramid. Each feature pyramid feeds two model evaluation processes: this is needed since the feature pyramid calculation is almost twice as fast as the model evaluation. Since the feature pyramids are independent of the model to be detected, each instance of the model evaluation block is able to search for another model, thereby reducing calculating time if one aims to detect multiple object classes at once.

Output. This final block gathers all the processed results and performs the post processing such as NMS (*non-maximum suppression*), reordering the frames, displaying detections and saving the frames.

Table 4.4: Speed results of our warp speed pedestrian detectors compared to the public implementation of the algorithm and fastHOG. We upscaled the image (×2), needed to detect pedestrians with a height around 50 pixels. No scene constraints are applied yet.

	640 × 480	1280 × 960	Speedup
LatSVMV4 (orig.)	1.33 FPS	0.33 FPS	1×
WSPD-v0.1 (float)	1.6 FPS	0.37 FPS	1.12×
WSPD-v0.2 (GPU)	2.97 FPS	0.85 FPS	2.58×
WSPD-v0.3 (hybrid)	12.9 FPS	4.17 FPS	12.7×
fastHOG	10.6 FPS	2.67 FPS	

Evaluation

We performed thorough experiments concerning both speed and accuracy of our implementations. Note that at this point all experiments are still performed without scene constraints (to be exploited using the warping window approach as described in the next subsection). The speed comparison of our and publicly available implementations is given in table 4.4.

All speed measurements were obtained on the same hardware (Intel Core i7 CPU 965 @ 3.20GHz with 12GB RAM and one nVidia GTX295 GPU). Our experiments were performed on both the original image size (640 × 480) as well as on an upscaled version (1280 × 960). This upscaling is needed if one wants to detect small pedestrians (around 50 pixels) since the HOG model is optimised for pedestrians with a height around 100 pixels. These upscaled image versions are used for our benchmark. We compared the relative speedup we obtained to the original algorithm and fastHOG [87], a publicly available fast implementation of the HOG algorithm [21]. As can be seen in the table, our C++ reimplementation with *floats* is slightly faster than the original implementation. With our implementation of the feature pyramid on GPU we achieve a speedup of 2.58×. While already faster, the overall speedup is limited since we only implemented the feature pyramid on GPU. Our multi-threaded hybrid CPU/GPU implementation achieves a speedup factor of 12.7×, thereby allowing real-time processing of 640 × 480 images (12.9 FPS).

The main motivation for using more advanced object detection algorithms in real-life applications is the increased accuracy. This however imposes a constraint on the allowed accuracy loss for fast implementations. Therefore we compared our reimplementations of the algorithm with existing algorithms on the publicly available Caltech dataset [34]. All experiments are performed on the challenging *reasonable* labelled pedestrians (as discussed in chapter 3,

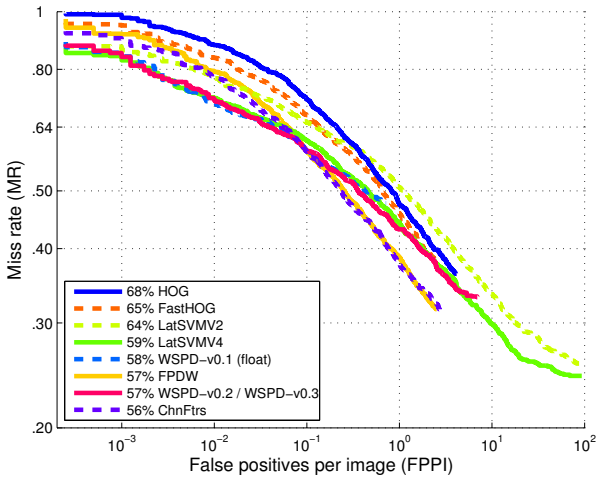


Figure 4.16: Comparison of existing detectors and our implementation. The results of HOG [21] and FPDW [31] are obtained from [35].

section 3.4). Our results are visualised in figure 4.16, where we display the miss rate (MR) versus the false positives per image (FPPI). In the legend of figure 4.16 the *log-average* miss rate is displayed to summarise the performance of each detector. This miss rate value is calculated by averaging nine FPPI rates spaced evenly in the log-space in the range of 10^{-2} to 10^{0}. See [35] for more details. The part-based detector (indicated with *LatSVMV4* [43] – *LatSVMV2* is an older release) achieves excellent accuracy results as compared to other state-of-the-art detectors (e.g. *FPDW* [31], *ChnFtrs* [32]), which was our motivation to use this detector as a baseline. These part-based detectors perform best on large-scale pedestrians [35]. However, detection accuracy loss on small pedestrians is minimal. Concerning our implementations, our speedup does clearly not come at the cost of a performance drop.

4.3.3 Accuracy improvement of the warping window approach

Apart from a fast implementation, a speed improvement is achievable by reducing the search space. Traditional object detectors use a sliding window paradigm, of which one of the bottlenecks is the large search space since at each position in the image every scale is evaluated. Often, ground plane assumptions are used for this matter [17]. They allow for a reduction in search space, while at the same time reducing the false positive rate. Such simple ground plane assumptions

(a) Example Caltech frame with labelled pedestrians

(b) Pedestrian height versus y-position

Figure 4.17: The warping window approach applied to the Caltech dataset

have their limitations though: they cannot cope with e.g. non-linear camera distortions. Therefore, to speedup detection we integrate our warping window approach that we introduced above (section 4.2) with our hybrid pedestrian detector to achieve higher processing speeds. This warping window approach can handle more complex scenarios, e.g. extreme camera viewpoints and wide-angle lens distortions. In this subsection we quantitatively show how we use the warping window approach to improve the accuracy on the Caltech dataset. In the next subsection (subsection 4.3.4) we then describe this integration and achieve extremely high detection speeds, even in the case of non-trivial camera viewpoints.

We now demonstrate how we can benefit from the warping window approach. We first illustrate the gain in accuracy on the Caltech benchmark dataset. Note that here we use the warping window as a post-processing step to show the potential accuracy improvement, whereas the warping window approach normally is used as a pre-processing step to reduce the search space. However, due to the basic camera viewpoint there, the full potential of the warping window is not exploited. As a reference, we start from the detection results of our multi-threaded hybrid pedestrian detector from subsection 4.3.2 (*WSPD-v0.3*) on the Caltech dataset. Here, due to the camera viewpoint no rotation is needed, hence we only model the pedestrian's scale at each image position. Since the camera is positioned in a forward-looking manner, we assume that at each horizontal pixel line the scale is constant, reducing the dimensionality of the lookup function (LUF) to one in this case. Based on the labelled Caltech training data from [35] (see figure 4.17 (a) for an example frame), we extract

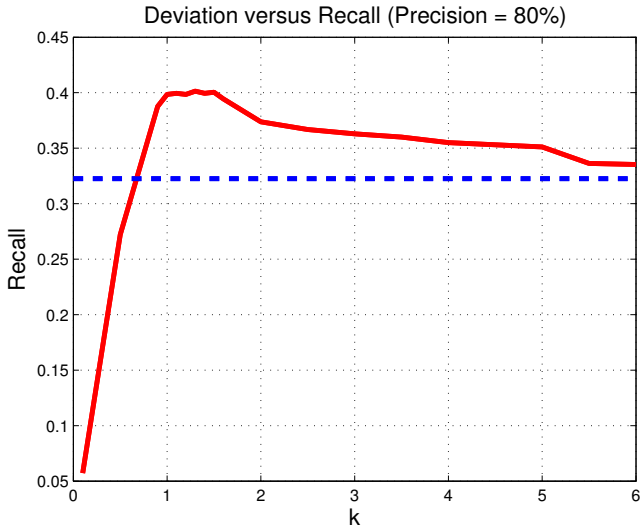


Figure 4.18: Influence of the k-value on the recall rate for a fixed value of the precision (80%).

the height of each labelled pedestrian (normalised with respect to 480 pixels, the image height) at each pixel position, and average those observations per horizontal pixel line. Annotations above the horizon are discarded. These data points are visualised in figure 4.17 (b). A linear relationship between the relative pedestrian height and the vertical pixel position is expected. Indeed, if the camera is mounted in such a way that the viewing direction is parallel to the ground plane, such a linear relationship should exist [100]. As seen, for low pedestrian heights non-linear behaviour is noticed. This is mainly due to errors in the labelling. For small pedestrians exact labelling is difficult, resulting in such annotation noise. Therefore, here we fit a third order polynomial function through these datapoints (solid red line). The dashed red lines illustrate two times the standard deviation (2σ) at each horizontal line. This transformation model can then be used to warp the ROIs to a fixed height.

However, as noted above, here we use it to prune the results to show the potential accuracy improvement. If the height of a new detection is *inconsistent* from what is expected at that particular position, the detection is discarded. As a measure of inconsistency, we use the degree of deviation. If we allow much deviation no or only slight improvements are obtained since little pruning is applied, while on the other hand limiting the possible deviation too much leads to a significant drop in the recall rate. We empirically determine the

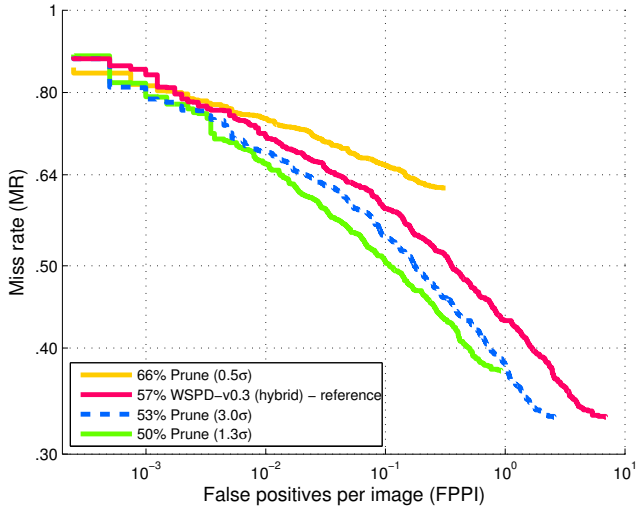


Figure 4.19: Accuracy improvement achieved on the Caltech dataset when using the warping window approach.

maximal allowed deviation (expressed as $k\sigma$) and evaluated the recall rate at a constant precision rate (80%). These results are displayed in figure 4.18. Here one clearly sees that the optimal deviation is found around 1.3σ, while at higher values of k the recall rate converges to the recall rate of the reference implementation (displayed as the dashed blue line). Figure 4.19 gives the miss rate versus the FPPI for both the original implementation along with three values of the deviation; an optimum is reached at 1.3σ. Applying the warping window approach thus leads to an accuracy improvement: at e.g. 0.1 FPPI, the miss rate decreases from 58% to 50%. In the next subsection we demonstrate how we use this approach to also increase the detection speed.

4.3.4 Warp speed pedestrian detector

In this subsection we now present the combination of the previously discussed warping window approach and the multi-threaded hybrid CPU/GPU implementation. As mentioned, this implementation achieves 12.9 FPS without the use of scene constraints. Here we demonstrate the integration of the warping window approach and propose our *Warp Speed Pedestrian Detector* (WSPD) which achieves pedestrian detection at up to 500 detections per second, without loss in accuracy.

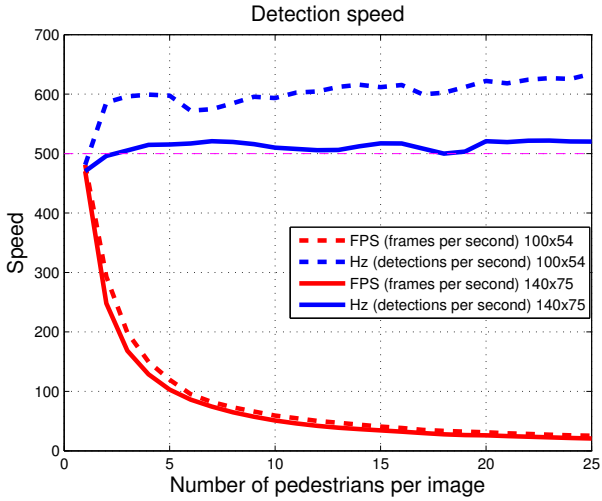
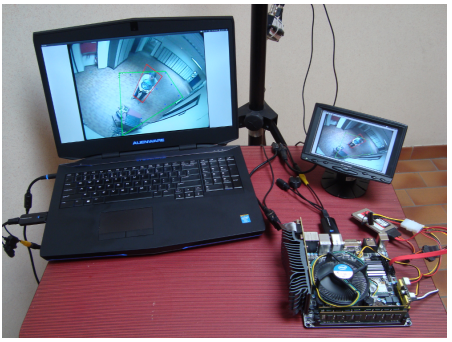


Figure 4.20: Speed results of our Warp Speed Pedestrian Detector for different ROI sizes.

In section 4.2 we described how the warping window approach is used to reduce the search space. At each pixel location only one scale needs to be evaluated. To further reduce the search space, we can use techniques that limit the number of locations where a detection needs to be performed, based on some detection probability measure. In a fixed scene, simple and efficient background modelling techniques (e.g. background subtraction) can be used. Such a scenario will be discussed in the next chapter. However, in moving scenes with a highly dynamical background more complex techniques are necessary. One such technique is tracking. If we are able to track the pedestrians throughout the scene, we can predict the search location, thereby reducing calculation time. Moreover we observe that in most applications the position where pedestrians enter the scene is predictable (e.g. doors). In the blind spot application discussed in section 4.2, the Kalman tracker [63] has proven to be a robust technique for this purpose. It predicts future positions based on detections in the past combined with a specific motion model. Based on this prediction, pedestrian ROIs are determined.

The crux of the matter is that we can use the warping window approach in combination with a Kalman tracker, and use the predicted search locations as input for our hybrid CPU/GPU pedestrian detector. This way we are able to perform pedestrian detection at unprecedented high speeds. Since we reduced the search space to a single scale and a single ROI, our detector only focuses on image content with high probability of containing pedestrians at a fixed

(a) Overview image of our demo with indicated blind spot zone



(b) A close-up view of our detection hardware

Figure 4.21: Overview of the live real-time demo that we developed based on our hybrid CPU/GPU detection framework.

scale, thus being very fast. The speed of our detector evidently depends on the size of the ROI(s) we extract from the source image. This influence is given in figure 4.20, where we give the speed of our detector for two ROI sizes. Both warping and detection times are taken into account here.

We display both the frame rate (FPS) and the number of pedestrian detections per second (Hz) we can achieve with respect to the number of pedestrians in the image. Initially with only one pedestrian in the image we achieve 480 FPS, using the 140 pixels high ROI as motivated in section 4.2. If there are more pedestrians per image our pipeline is used more efficiently, thus the number of detections per second increases. However, this evidently leads to a decrease in the number of frames per second. At e.g. 20 pedestrians per image we still achieve an impressive frame rate of 25 FPS.

4.3.5 Real-time demonstrator

Using the hybrid implementation discussed above we developed a live real-time demonstrator of our complete blind spot detection system. This demo was presented at the *Vision & Robotics Fair* in 2013 and the *AAA Vision Symposium* in 2014. Figure 4.21 (a) displays an overview image of our demo. In figure 4.21 (b) a close-up of the hardware setup is given (details are discussed below). We altered our original hybrid implementation as follows. A genuine blind spot camera is used as input, mounted at approximately the same height as employed on our test truck. First, we defined a zone in the image in which

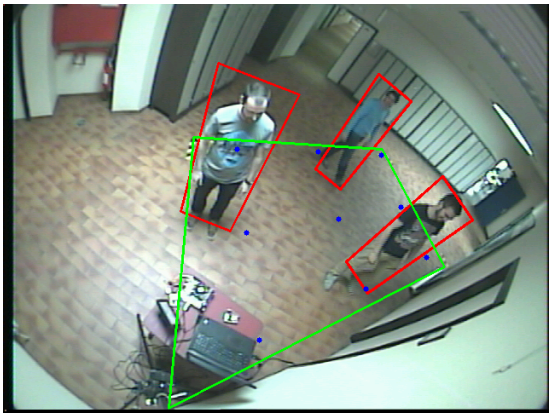


Figure 4.22: An example frame of our demonstrator. The green zone indicates the simulated blind spot area. Search points are drawn in blue.

Table 4.5: Overview of the different hardware platforms used in our demonstrator.

	CPU (Intel)	**GPU (nVidia)**
Desktop PC	Xeon E5-2687W @ 3.1 GHz	Quadro 2000
Laptop	i7-4710MQ @ 2.5 GHz	GeForce GTX880M
Embedded System	i7-4770S @ 3.1 GHz	GeForce GT610

pedestrians ought to be detected. This zone effectively corresponds to the blind spot zone of a truck. Next, in this blind spot zone we homogeneously positioned several search coordinates to maximally cover the complete search zone. This search grid replaces the initial search coordinates used above. We eliminate the tracking framework, and perform a frame per frame detection. Each of these grid points then serve as input for our warping window framework. We thus effectively warp each region defined by these grid points, and evaluate a pedestrian detector on each warped image patch. Figure 4.22 displays a typical input frame with detections. The green region indicates the simulated blind spot zone, and the blue points display the search grid.

We evaluated four different grid sizes: 3×3, 4×3, 4×4 and a 5×5 grid. These grids are displayed in figure 4.23. Evidently, a denser grid results in a higher accuracy at the cost of a higher computational complexity. In this section we give speed measurements for all grid sizes. The drop in accuracy when using these grids is explored in chapter 8, section 8.2. We implemented this demo on three different hardware platforms; a high-end desktop computer, a high-

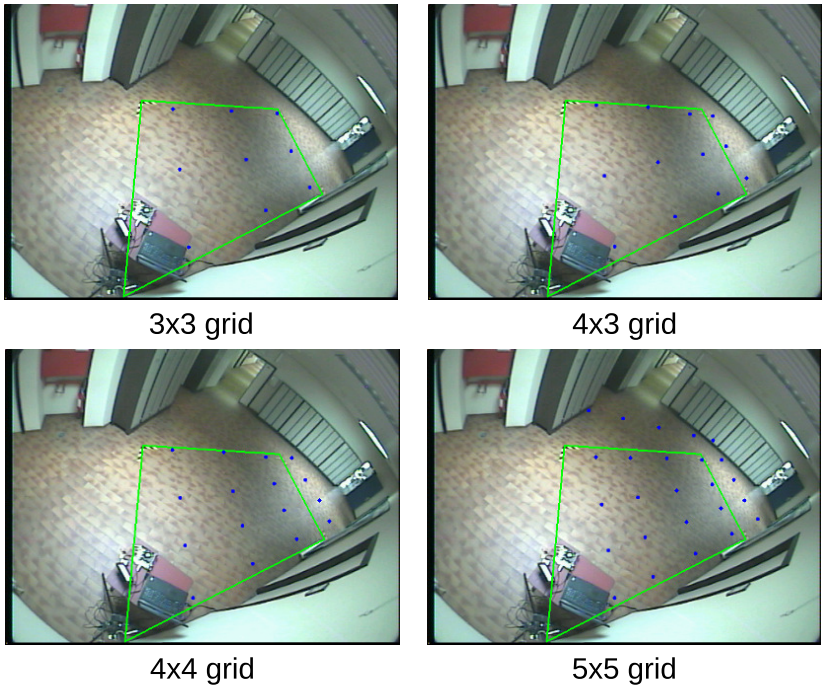


Figure 4.23: Overview of the different detection grids.

end laptop and finally an embedded platform attached to a small automotive monitor, as seen in figure 4.21. This embedded platform evidently is easily integratable in a real truck. The hardware specifications for each platform is given in table 4.5.

An identical implementation was used on all three platforms. No optimisation was performed with respect to the number of threads. We utilised two feature pyramid threads, and four evaluation threads (similar to figure 4.15). Figure 4.24 displays the speeds results on all three platforms for different grid sizes. The speed difference between the high-end desktop computer and the laptop is minimal. We achieve a maximum of 420 Hz (16.9 FPS with the 5×5 grid) on the desktop computer. For the embedded platform, on average a detection speed of about 140 Hz is reached. For the 3×3 grid we achieve a detection speed of 15.1 frames per second, and thus achieve real-time performance (our blind spot camera output frames at 15 FPS). Evidently, as the grid size increases the processing speed decreases. For both the desktop computer and laptop this is only noticeable at higher grid size, indicating that our software does not fully exploit the hardware capabilities on these platforms. For example, the number

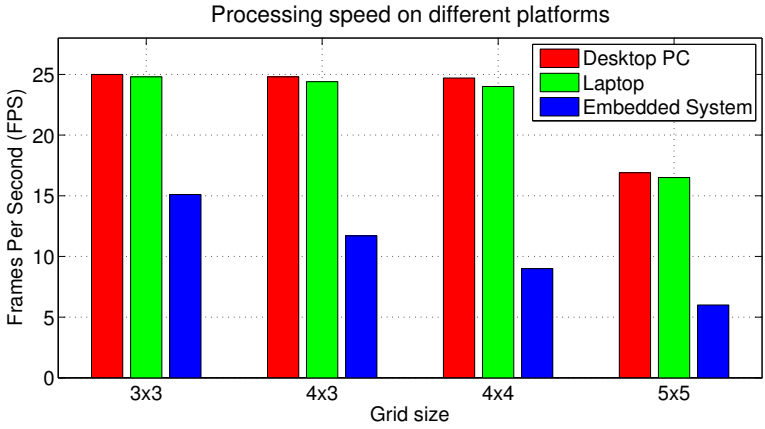


Figure 4.24: The processing speed of our demonstrator for multiple platforms and different grid sizes.

of threads could be matched with the specific hardware. An increase of the grid size on the embedded platform is immediately visible, as the processing speed significantly drops. However, as we will show in chapter 8, even with a small grid size excellent accuracy results are obtained.

4.3.6 Conclusion

In this section we presented the combination a fast hybrid CPU/GPU implementation and the exploitation of scene constraints, resulting in our WSPD. Our hybrid detection algorithm combined with a reduction of the search space allows for pedestrian detection at 500 detections per second. Experiments concerning both accuracy and speed on the Caltech dataset show that this speedup does not lead to a decrease in accuracy. Furthermore we presented accuracy improvement measures when applying our warping window approach on a well-known dataset, and integrated this approach with our WSPD. Finally, based on this implementation we developed a live real-time demonstrator and presented speed measurements on multiple platforms. We showed that – even on an embedded system – we achieve real-time processing speeds.

4.4 Conclusion

In this chapter we discussed a first step towards a fast and accurate blind spot detection system. We presented an efficient pedestrian tracking-by-detection framework targeting the challenging blind spot camera images. This chapter was subdivided into three main parts.

In the first part of this chapter we introduced an initial tracking-by-detection framework developed for standard backward-looking eye-level images recorded from a driving vehicle. For this, we implemented and compared two traditional tracking approaches and discussed their effectiveness for our application. This tracking framework served as a baseline for subsequent sections in this chapter.

The second part of this chapter then introduced our warping window approach. This approach allows for the efficient detection of pedestrians in the highly distorted images of our blind spot camera. Since to the best of our knowledge currently no such datasets exist, we recorded our own a dataset using a genuine blind spot camera and real truck. For this, we simulated several dangerous blind spot situations involving pedestrians, yielding a unique dataset. Extensive speed and accuracy experiments were performed on this dataset, indicating that both excellent speed and accuracy results are obtained.

The final section of this chapter focused on an efficient implementation of this warping window approach. We presented a hybrid CPU/GPU optimised implementation which achieves detection rates of up to 500 Hz. We showed that, even on an embedded platform which is easily implementable in a real truck, real-time processing speeds are achieved. A live real-time demonstrator has been developed based on this implementation. Additionally, in this section we presented experiments which indicate the gain in accuracy that is achieved when using our warping window approach on a publicly available dataset.

However, note that the pedestrian detection framework presented here is only an initial approach. Further refinements are needed. Indeed, currently only pedestrians are detected. Therefore, in the next chapters we present several methodologies which further enhance our detection approach. For this, we present improvements concerning both accuracy and the inclusion of additional object classes (such as bicyclists).

In the next chapter we show that the warping window approach developed here is indeed generalisable to other applications where similar camera viewpoints occur. For this, we employ a typical surveillance scenario.

Chapter 5

Pedestrian detection in challenging surveillance videos

In this chapter we show that our warping window approach presented in the previous chapter easily generalises to other scenarios where non-standard camera viewpoints occur. Therefore, apart from the blind spot application previously discussed, we apply our approach to a surveillance scenario. This work was published at the VISAPP 2015 conference [109]. A revised and extended version was published as a book chapter in CVICGTA 2016 [111].

Typical surveillance images are challenging to analyse since the overall image quality is low (e.g. low resolution and high compression). Furthermore often birds-eye viewpoint wide-angle lenses are used to achieve maximum coverage with a minimal amount of cameras. These specific viewpoints make it unfeasible to directly apply existing pedestrian detection techniques. Moreover, real-time processing speeds are required.

To overcome these problems we apply the previous chapter's technique on these challenging surveillance images. These typical surveillance scenarios employ fixed cameras. As such, in this chapter we are able to integrate foreground segmentation techniques in our previously proposed warping window approach. We performed extensive experiments on publicly available real-life video sequences. Our approach achieves excellent accuracy results while still meeting the stringent real-time demands needed for these surveillance applications, using only a single-core CPU implementation.

5.1 Introduction

Reliable pedestrian detection and tracking in surveillance images opens up a wide variety of applications (e.g. abnormal behaviour detection, path prediction, intruder detection, people safety on e.g. movable bridges and crowd counting). In recent years, tremendous advances concerning pedestrian detection were published. Current state-of-the-art detectors achieve excellent accuracy results on publicly available datasets (see chapter 3, section 3.3). Unfortunately, directly applying these existing techniques on challenging surveillance images is not a trivial task. This is due to the inherent nature of these surveillance applications; often a large number of cameras are utilised since large areas need to be covered completely. Such scenarios impose severe constraints on the hardware: low-cost cameras are employed with wide-angle lenses, mounted high in a partially down-looking birds-eye view. Consequently image processing and analysis on these images is challenging.

Indeed, typical surveillance images are often captured at low-resolution and use high compression. Classic background subtraction based object detection methods yield very noisy results at these high compression ratios. Moreover, these techniques do not differentiate between people and other objects. Due to their specific viewpoint (and wide-angle lens) standard pedestrian detectors - which are trained and evaluated on forward-looking images - are also unable to give accurate detection results on these images. Additionally, due to perspective effects some pedestrians to be detected appear very small in the image, which remains one of the most challenging tasks for current pedestrian detectors [35]. Furthermore, real-time processing speeds are required. In this chapter we propose a flexible and fast pedestrian detection and tracking framework specifically addressing these challenging surveillance images. See figure 5.1 for a typical example frame of the publicly available surveillance dataset we used [14].

Our approach achieves excellent accuracy results at real-time processing speeds. We overcome the above mentioned challenges by the integration of three modalities: foreground segmentation approaches, the exploitation of scene constraints and an accurate pedestrian detector. This is done as follows. First, candidate regions in the image are generated. Using a calibrated scene distortion model, an early rejection of false patches is achieved. Next the candidate regions are warped to a standard viewing angle and used as input for a state-of-the-art pedestrian detector. As explained in section 5.3 our approach allows for the use of a highly accurate pedestrian detector which would otherwise be too computationally intensive for real-time applications. Finally, the detections are employed in a *tracking-by-detection* approach to further increase the accuracy. Note that using our approach the actual scene calibration is trivial and easily

Figure 5.1: Example frame of one of the sequences of the CAVIAR dataset [14].

performed. We demonstrate the effectiveness of our approach on challenging surveillance video sequences, and present extensive accuracy and speed results. Our approach is generalisable to other object classes. The remainder of this chapter is structured as follows. In section 5.2 we discuss related work on this topic, and distinguish our approach from existing work. Section 5.3 presents our framework in detail. Next we propose experimental results on challenging sequences in section 5.4. Finally, section 5.5 concludes this chapter.

5.2 Related work

An extensive overview of related work concerning pedestrian detection was already discussed in chapter 3. Therefore, in this section we only discuss related work concerning these specific surveillance scenarios.

Several pedestrian tracking algorithms exist. Due to recent advances in object detection techniques, as mentioned tracking-by-detection has become increasingly popular. There, an object detector is combined with a reliable tracking algorithm (e.g. particle filtering); see for example [12]. Existing work on pedestrian tracking in surveillance images mainly focuses on standard viewpoint and/or high-resolution images [99]. For example, in [10] the authors present a tracking framework relying on head detection. For this, a significantly high resolution is required. Other approaches employ thermal cameras to facilitate segmentation to reduce the search area [54, 68, 69].

In the previous chapter we presented a real-time pedestrian detection framework for similar viewpoint images which are captured with a blind spot camera mounted on a real truck. These images are - apart from the viewpoint - challenging since the camera is moving. However, in this work we can fully exploit and integrate foreground segmentation methods to increase both accuracy and speed. Furthermore, we work with images captured from genuine surveillance cameras. These images are of low-resolution, low-quality and, due to the use of wide-angle lenses show large amounts of distortion and contain non-trivial viewpoints.

Existing work on the same dataset either employs clustering algorithms with GPU optimisation [79], or focuses on motion analysis by matching trained silhouette models [91, 92]. We differ significantly from these previous works: we developed an accurate tracking framework in which we can employ a highly accurate pedestrian detector on these challenging images, and thus perform much better than existing methods. We achieve real-time processing speeds on a single-core CPU implementation. Our approach easily lends itself for multi-threaded implementation if higher computational speeds are needed.

5.3 Algorithm overview

Running standard pedestrian detectors such as the *Deformable Part Models* on surveillance images as shown in figure 5.1 is unfeasible. Current pedestrian detectors are only trained on upright pedestrians at a fixed height. Scale invariance is achieved using a scale-space pyramid. Thus in order to achieve decent detections on these surveillance images the detectors ought to run on multiple rotations and scales of the same surveillance image, using both dense rotation and scale steps. Evaluating the total 4D rotation-scale search space in real-time evidently is impossible. Nonetheless, the use of a pedestrian detector could significantly increase the accuracy, as opposed to standard techniques which only rely on e.g. background subtraction with blob analysis due to time constraints.

Therefore, to overcome these challenges we propose the integration of a foreground segmentation approach with a scene model and a highly accurate pedestrian detector. Our approach allows for the detection of pedestrians in challenging (e.g. rotated and perspectively distorted) viewpoints under large lens distortion at low computational complexity, with very high accuracy. To retrieve the scene model, a simple one-time calibration procedure is performed, no explicit lens or camera calibration is needed. Our algorithm briefly works as follows. As seen in figure 5.1, pedestrians appear rotated and scaled based on their position

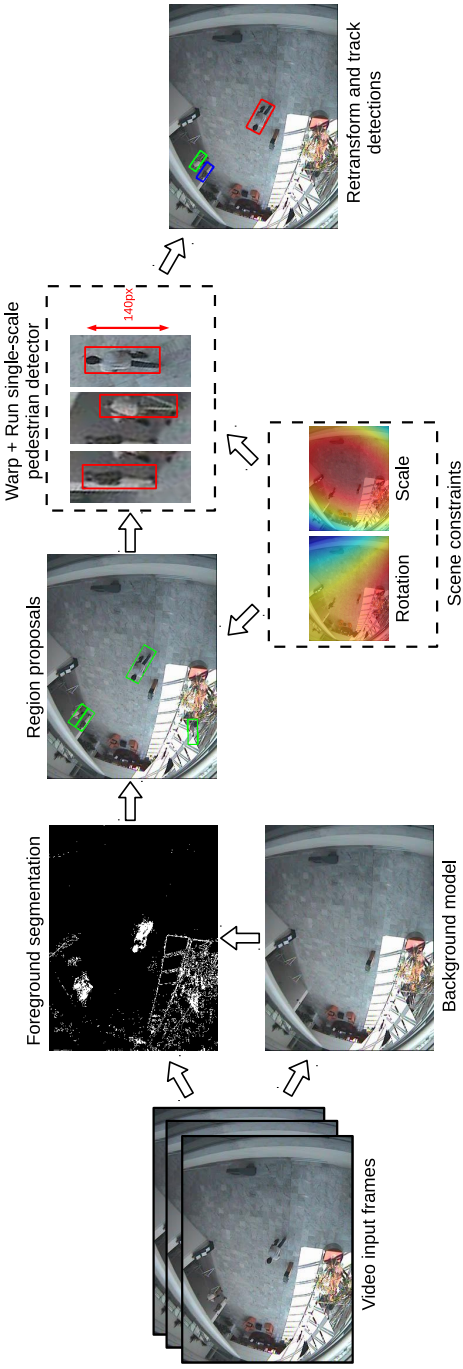


Figure 5.2: Overview of our detection pipeline. After a first foreground segmentation step we extract region proposals which potentially contain pedestrians. Each region is warped to an upright fixed-height patch. Next, a highly accurate pedestrian detector is evaluated at a single scale. Finally, the detections are retransformed and tracked.

in the image. We exploit this scene knowledge throughout our detection and tracking pipeline. For each input image, after a preliminary segmentation, we generate region proposals which potentially contain pedestrians. The scene model is used to reduce the number of region proposals. Next, based on the position in the image we warp each valid potential region to an upright and fixed-height image patch using our warping window approach from chapter 4. These patches are given as input to a state-of-the-art pedestrian detector, which evaluates a pedestrian model on a single scale only.

This is the key advantage of our work: since only one scale and position needs to be evaluated we can use a highly accurate pedestrian detector which would otherwise be too time-consuming. Furthermore this approach allows for the detection of extremely small pedestrians, if the detection model is powerful enough. The detections are retransformed to the original input image, and employed in a tracking-by-detection framework to associate pedestrian tracks and handle missing detections. Since each region can be evaluated independently, a fast multi-threaded implementation of this approach is trivial. Figure 5.2 shows an overview of our approach. In the next subsections we describe further details of each step in our pipeline, and motivate important design choices.

5.3.1 Foreground segmentation

First we perform a foreground segmentation step to identify moving regions in the static camera images. Several segmentation approaches are applicable ranging from basic background subtraction methods to more advanced motion estimation methods. Since we employ scene constraints further on to reduce the number of region proposals, our approach allows for the use of a coarse segmentation. For this step we thus prefer low computational complexity over high accuracy, excluding time-consuming techniques (e.g. optical flow). Hence, we rely on background estimation techniques which generate a statistical model of the scene.

Several popular methods exist. Since a comprehensive comparison of these techniques is out of the scope of this work, we refer to [9] and [82] for a detailed overview. Concerning background subtraction, the main challenges in typical surveillance images arise from changing lighting conditions and camera shake. Based on these comparative works we opted for the method of [129], which employs *Gaussian Mixture Models* (GMM). These methods have proven to cope well with (limited) background motion.

Their proposed method is an extension of the original GMM where the number of Gaussian components per pixel is automatically selected. This effectively reduces memory requirements and increases the computation speed, making it

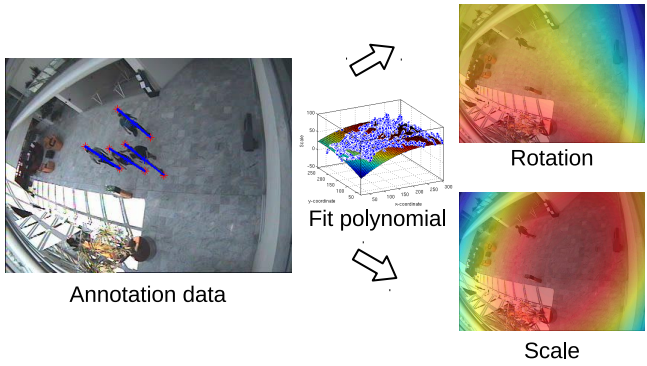


Figure 5.3: A one-time calibration step is needed. The transformation parameters are extracted from the annotations.

ideal for this application. A qualitative segmentation output example is shown in the overview figure (figure 5.2).

5.3.2 Modelling scene constraints

To model the scene constraints, we apply our warping window approach as presented in the previous chapter. As mentioned, the pedestrians in the surveillance images appear rotated and scaled. Since the position of the surveillance camera is fixed with respect to the ground plane both parameters only depend on the position in the image. If we know the rotation and average pedestrian height for each pixel position we can exploit this scene knowledge to achieve fast and accurate pedestrian detection. During the generation of the region proposals this information can be used to reject regions that diverge too much from the expected region properties, thus limiting the search regions. For each valid proposed region, we use the transformation parameters to warp each patch to an upright, fixed scale image patch, allowing the use of an accurate pedestrian detector while being real-time. To retrieve these transformation parameters, again a one-time offline calibration needs to be performed (see figure 5.3). However, the scene calibration as proposed here is easy to perform and trivial. For this, we extracted the rotation and height of each annotated pedestrian from the dataset, giving the scale and rotation for that specific point. As in the previous chapter, we interpolated these data points using a second order 2D polynomial function $f_i(\mathbf{x})$ for both parameters. Both $f_{scale}(\mathbf{x})$ and $f_{rotation}(\mathbf{x})$ are used as *Lookup functions* (LUFs): at each position in the image they define the expected region properties and transformation parameters.

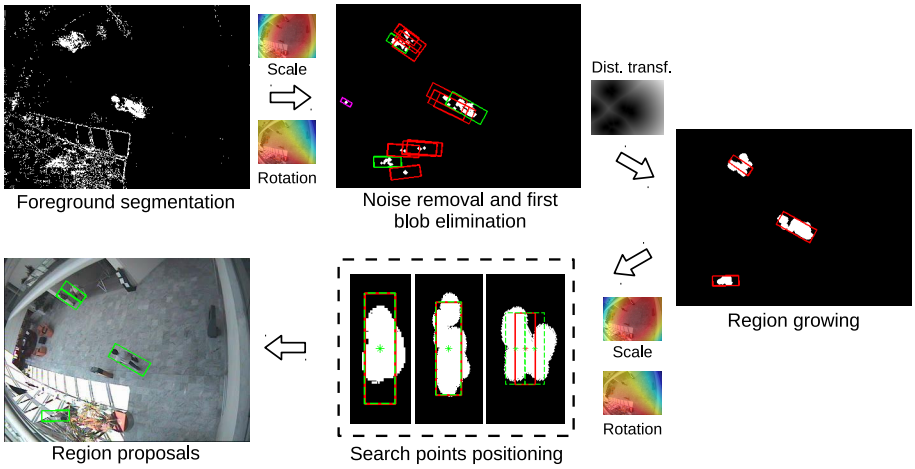


Figure 5.4: Our region proposals pipeline. After foreground segmentation and noise removal a first blob elimination is performed. Next we perform region growing using a distance transform. Finally, we determine the optimal search points.

5.3.3 Generation of region proposals

In a next step we refine the segmentation and generate region proposals which need to be warped and evaluated using our single-scale pedestrian detector. Since we employ a pedestrian detector in the next stage to validate each region we are allowed to propose more regions than needed, i.e. regions without pedestrians. An accurate detector should indeed negatively classify such patches. However, it is important to early reject false patches, since they lead to useless computations and lower processing speeds. This stage thus tries to balance between optimal accuracy and speed, generating an optimal amount of search locations. Figure 5.4 gives an overview of our region proposal calculation. We now discuss each consecutive step in this pipeline.

First elimination. As a preprocessing step, we first eliminate noise in the segmentation which remained after the background subtraction step (due to e.g. changing lighting conditions). This is simply done using *morphological opening*. Next, we perform a connected component analysis (using 8-connectivity), and test the local scene model for each blob. That is, we construct a bounding box of the expected scale and rotation around the centroid of each blob. We reject two types of regions: extremely small ones (25 pixels or less, drawn in magenta

in the second step of figure 5.4), and those that diverge from an area constraint (drawn in red). For this constraint, we require that the area of the connected component should be larger than a minimal percentage of the expected area (15%). This step eliminates most invalid regions.

Region growing. In the case of insufficient contrast, the foreground segmentation performs suboptimally (i.e. tends to split a valid pedestrian in multiple blobs, as seen for the largest pedestrian in figure 5.4). For each remaining valid region we therefore perform region growing based on the Euclidean distance transform, joining regions nearby. This has a second advantage: multiple pedestrians which are nearby are joined into a single detection region, even if one of them was removed after the first elimination. This is also illustrated in figure 5.4: after the first elimination only one of both small pedestrians is maintained. However, after region growing both are connected.

Defining search points. Finally, we define exact search locations where the pedestrian detector will be applied. This is done as follows. Each remaining region is again verified against the scene constraints since, due to the previous step, these regions could have grown significantly. This is the case when multiple (possibly previously invalid) regions are joined. Note that we do not reject regions at this stage. We locally evaluate each region and use the expected height and rotation to estimate the number of possible pedestrians. Based on the size of the region we first evaluate if multiple search points are needed for this region. If so, we define a linear grid over the entire region of which the step size depends on the ratio of the expected and actual region parameters, and eliminate grid points which are located outside the segmented region.

The final region proposals are visualised as the green rectangles shown in the bottom left image in figure 5.4. As seen, our regions accurately predict possible pedestrians in the image. This is the power of this approach: by combining foreground segmentation and scene model constraints the search space for the computationally expensive pedestrian detector can be enormously restricted. Slight deviations from the exact pedestrian position are allowed since we employ a sliding-window approach in the final warped patch.

5.3.4 Warping patches

Our scene model has another advantage: for each image location we know how a pedestrian is locally distorted. Each region proposal is warped to an upright

pedestrian at a fixed scale. Using this approach we are able to accurately detect even rotated and extremely small pedestrians, using a single-scale pedestrian detector only. The region proposals I are warped such that $I_{warp} = TI$ where transformation matrix T simply consists of a Euclidean transformation of which the parameters are extracted from the LUFs. Note that the optimal scale to which the patches are warped highly depends on which pedestrian detector is used. This is discussed in the next section, where we motivate the choice of pedestrian detector and determine the optimal scale.

5.3.5 Pedestrian detector

The warped image patches are now classified by a pedestrian detector. In fact, the method described in the previous sections is generic and can be combined with each existing pedestrian detection algorithm. To select the most appropriate pedestrian detector to use in this framework, similar arguments as given in chapter 4, section 4.2 are applicable. As discussed in chapter 3, recent R-CNN based detection methods currently achieve top accuracy results concerning object detection in general. However, their performance is far from real-time, and they are more suited for multiclass large database retrieval tasks. Rigid pedestrian detectors (such as ChnFtrs) currently offer the best trade-off between speed and accuracy when a full-scale space pyramid needs to be constructed. However, since we need to evaluate a single scale only, no scale-space pyramid needs to be constructed.

Therefore we are able to use an accurate pedestrian detector which would otherwise be too time-consuming, such as the *Deformable Part Models*. Moreover, since a rigid model does not allow for any deformation, using it in our single-scale approach is even unfeasible in a direct manner. Since natural slight height variations exist between pedestrians, the detection accuracy significantly drops when using these models on a single-scale. Given this information, we opted to use the cascaded DPM model [43]. We again altered this detector into a single-scale only implementation and performed experiments to determine the optimal scale factor to which the region proposals need to be warped. In the previous chapter we performed similar experiments to determine the optimal upscale factor of the DPM pedestrian detector.

However, here we repeat similar experiments, since the specific scale factor might be dataset dependent due to small calibration and labelling errors. Apart from the optimal scale factor, in these experiments we simultaneously determine the optimal detection threshold. This is done as follows. We randomly extracted about 6000 annotated pedestrians from the CAVIAR dataset and warped them to different scales (heights). Combined with 6000 negative patches we calculated

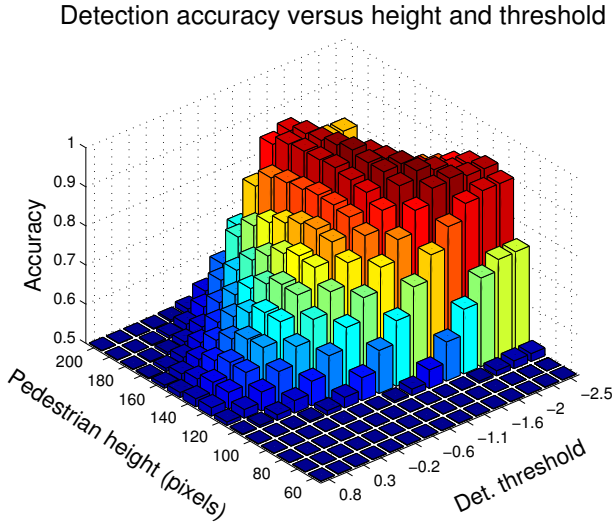


Figure 5.5: The accuracy versus the pedestrian height and detection threshold for the single-scale cascaded DPM detector.

the accuracy in function of the height and detection score threshold. This is done as follows. All patches are evaluated – for different heights and different detection thresholds – using our DPM pedestrian detector. Each patch is classified and assigned as being either a TP, FP, TN of FN. We calculate the accuracy as:

$$Accuracy = \frac{TP + TN}{P + N} \tag{5.1}$$

Where $P = TP + FN$ and $N = TN + FP$. These results are shown in figure 5.5. As can be seen, at low resolutions the accuracy drops significantly, since only very limited spatial information is available. At high resolution similar behaviour is seen, since the pedestrians mismatch the detection model. Concerning the detection threshold, the detection accuracy is low at both high values (high *false negative rate*) and low values (high *false positive rate*).

Figure 5.6 displays the optimal threshold slice extracted from figure 5.5. The accuracy is almost constant between 130 – 170 pixels. However, at larger pedestrian heights the detection time significantly increases. We therefore used 140 pixels as our optimal rescale height to which the region proposals will be warped such that a one-scale pedestrian model can be directly applied.

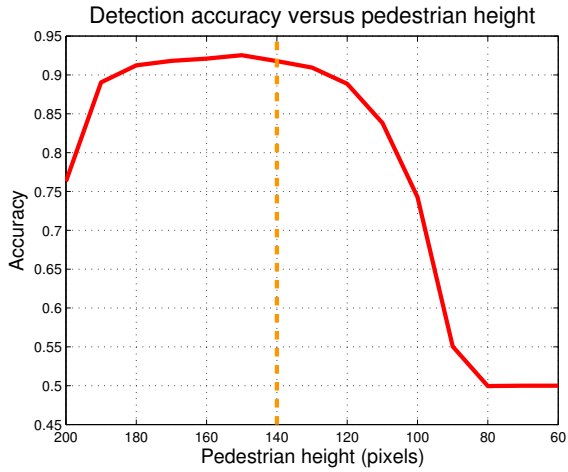


Figure 5.6: The optimal threshold slice displaying the accuracy versus the pedestrian height.

5.3.6 Tracking

The resulting detections are then retransformed to the input image coordinates. Next a *non-maxima suppression* step is performed, in which overlapping detections are filtered; only the highest scoring detection is kept. To link detections over multiple frames and to cope with occasional missing detections we integrate our approach in a tracking-by-detection framework, almost identical as in the previous chapter. There, the tracking framework uses initial search coordinates strategically placed at locations in the image where pedestrians are expected to enter the frame. If a pedestrian is detected in these initial search regions – which are always evaluated – a track is started and the predicted location of the pedestrian in the next frame is used (together with the initial search coordinates) as input to the *warping window* approach.

In this chapter we step away from the use of initial search coordinates. Indeed, here the *region proposals* (based on the foreground segmentation) are used as input for the warp and detection stage. However, the details of the tracking framework (e.g. the state vector and transition matrix) remain the same. We thus omit implementation details here, and refer the reader to chapter 4, subsection 4.2.2. As a reminder, we employ a Kalman tracker with constant velocity motion model. See figure 5.7 for two qualitative tracking sequences of our proposed algorithm.

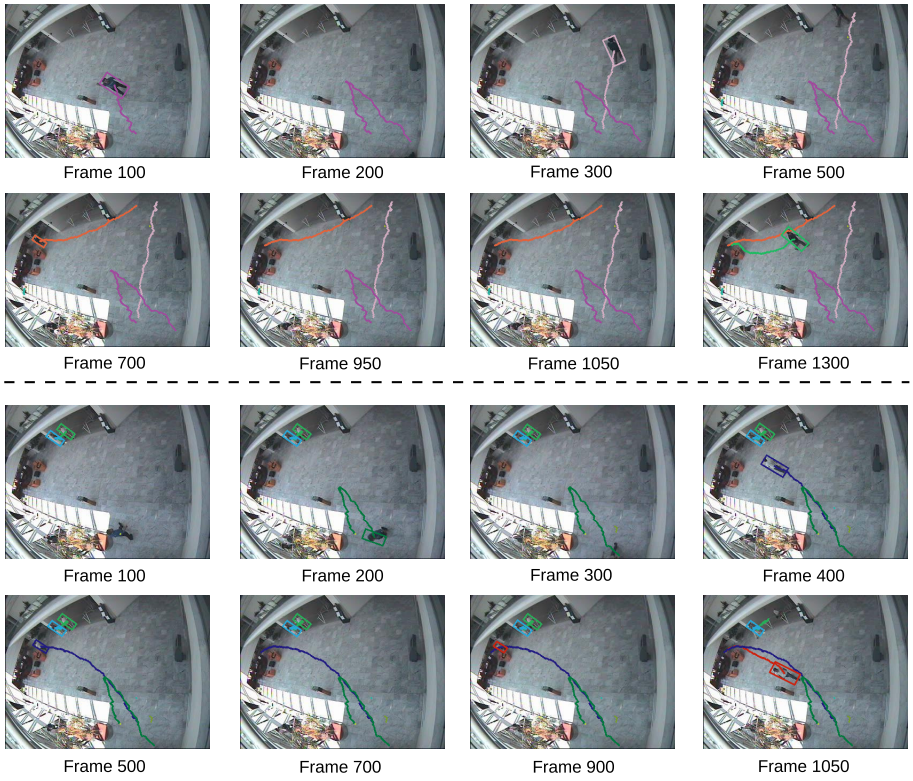


Figure 5.7: Qualitative tracking example on two of the evaluation sequences. See `http://youtu.be/kWoKBPQoeQI` for a video.

5.4 Experiments and results

We performed extensive experiments concerning both speed and accuracy on the publicly available CAVIAR dataset [14]. This dataset was recorded at the entrance lobby of the INRIA labs with a wide-angle camera lens. The images are taken with a resolution of 384×288 at 25 frames per second, and are compressed using MPEG2. See figure 5.1 for an example frame. The dataset is divided into six different scenarios: *walk*, *browse*, *meet*, *leave bags*, *rest* and *fight*. Each scenario is again subdivided into multiple sequences, making a total of 28 sequences. We used all sets for testing.

Note that some sequences contain pedestrians which are inherently undetectable with our proposed framework. For example, the *fight* sequences include scenarios with people in specific fighting poses, and the *rest* sequences contain scenarios

Table 5.1: Overview of the different sequences of the CAVIAR dataset.

Scenario	**#frames**	**diff.**	**comments**
Walk	3045	easy	Few people, low interaction.
Browse	6654	medium	People browse at e.g. reception desk.
Leave bags	5839	medium	Leaving objects behind.
Rest	4220	medium	Resting on floor and in chairs.
Fight	2492	difficult	People fighting. Difficult poses.
Meet	4123	difficult	Group meetings, multiple occlusions.

where people fall on the floor or rest in e.g. chairs thus violating our scene constraints. Table 5.1 gives a textual overview of each scenario. For each scenario we give a *difficulty* measure, i.e. an indication of the complexity of the sequences of each scenario. Easy scenarios are composed of simple sequences with only few people and low interaction whereas difficult scenarios contain many occlusions and challenging poses. In total, our evaluation set consists of about 26400 frames, containing about 36200 annotations.

Our algorithm is implemented in Matlab, with time-consuming parts (e.g. the detection and transformation) in C and OpenCV (using *mexopencv* as interface). Our test hardware consists of an Intel Xeon E5 CPU running at 3.1 GHz. All speed test are performed on a single CPU core. However, a multi-threaded CPU implementation to further increase the processing speed is trivial.

5.4.1 Accuracy

Figure 5.8 and figure 5.9 display the accuracy results of our algorithm, using *precision-recall* curves. We give results for all scenarios mentioned above, ranging from easy (e.g. *walk* - limited number of persons) through difficult (e.g. *meet* - multiple persons with occlusions). For the sake of clarity we spread the accuracy results of the six sequences over two separate plots, based on their difficulty. The accuracy plot of figure 5.8 groups the easy and medium scenarios while figure 5.9 gives the accuracy results for the more difficult scenarios.

We exclude small pedestrians from the annotations (smaller than 20 pixels), and remove annotations in the top left corner of the image (on the balcony) and the bottom left corner of the image (people behind the covered reception desk). Furthermore we discard annotations close to the image border, since the pedestrians are not completely visible there (the annotation is strict and already starts when part of a pedestrian enters the frame). The solid lines in figure 5.8 and figure 5.9 indicate the accuracy without tracking, whereas

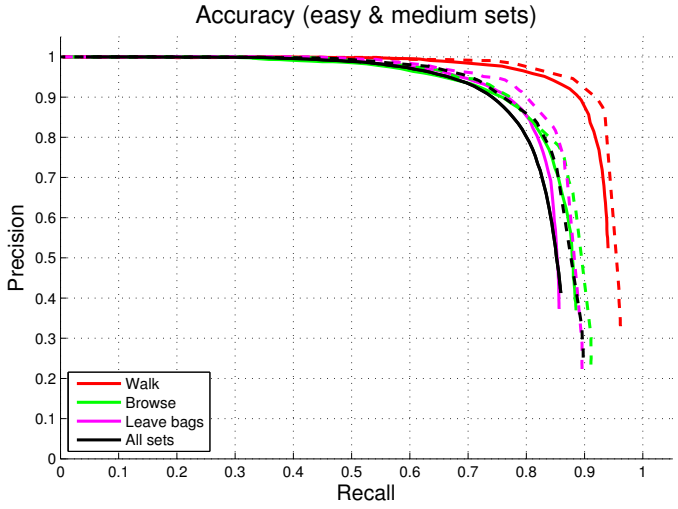


Figure 5.8: The accuracy of our algorithm over the easy and medium sets of the CAVIAR dataset. Solid lines indicate the results without tracking, dashed lines include tracking. The black curve (*All sets*) indicates the average accuracy over all six scenarios.

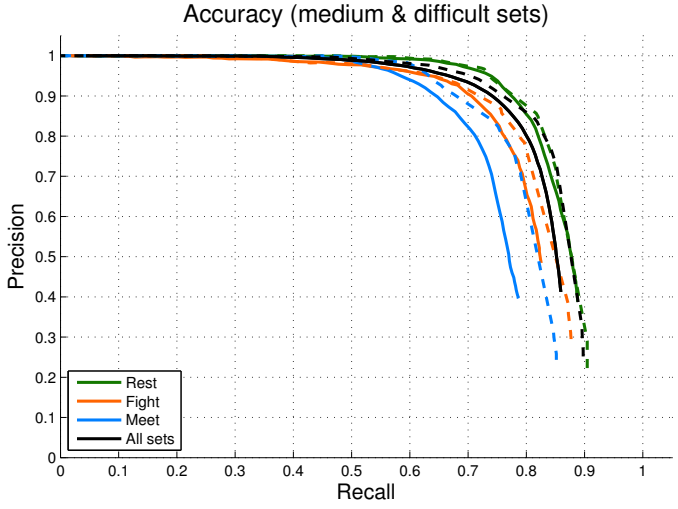


Figure 5.9: The accuracy of our algorithm over the medium and difficult sets of the CAVIAR dataset. Solid lines indicate the results without tracking, dashed lines include tracking. The black curve (*All sets*) indicates the average accuracy over all six scenarios.

Figure 5.10: Example of warped annotations. Low-resolution and high-compression artifacts are noticeable.

the dashed lines show the accuracy with tracking. The black curves on both figures indicate the total average accuracy over the entire evaluation set (all six scenarios). To indicate the difficulty, in figure 5.10 we display some extracted annotations which are warped to a fixed scale and upright position.

As can be seen, these low-resolution output images contain severe compression artifacts. Even for humans they are sometimes difficult to recognize as a pedestrian. However, we achieve excellent accuracy results given these strict dataset annotations and challenging nature of these images. As observed, on some difficult scenarios (e.g. *Meet* and *Rest*) a lower accuracy is obtained. This is mainly due to two reasons: these sets contain many long-term occlusions and poses that a standard pedestrian detector is unable to detect (e.g. sitting in a chair, lying on the floor). Since our tracker handles missing detections, the accuracy significantly improves.

5.4.2 Speed

The exact calculation time depends on the number of region proposals per image. Figure 5.11 therefore displays the speed of our algorithm (in frames per second), versus the number of region proposals. Evidently, the processing speed decreases when multiple region proposals need to be evaluated. However, even at e.g. four region proposals we still achieve 17 FPS.

Over the entire evaluation set we achieve an average of 32 frames per second, indicated with the dashed red line. Note that all experimental results are performed on a single CPU core. In fact, each region proposal can be evaluated independently, thus allowing for an easy multi-threaded implementation. Figure 5.12 visualises the individual calculation times for each important step

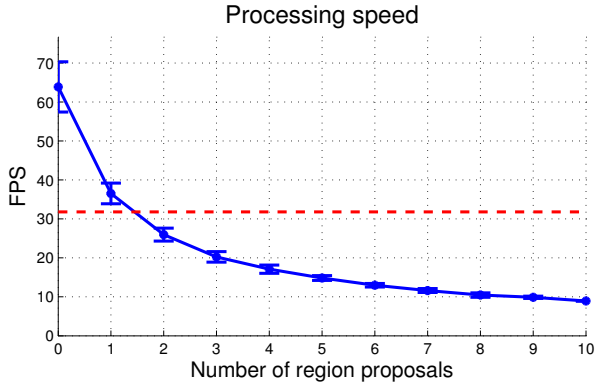


Figure 5.11: The processing speed of our algorithm versus the number of region proposals.

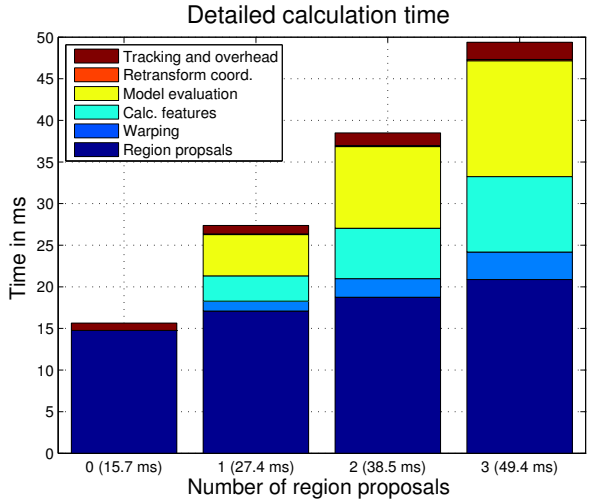


Figure 5.12: An overview of the calculation time for each step in the algorithm versus the number of region proposals.

in the entire algorithm pipeline, for a varying number of region proposals. As visualised, generation of the region proposals takes about 15-20 ms. The warping operation is very fast: on average 1 ms per region proposal is needed. Concerning the pedestrian evaluation step, the average feature calculation time per region is about 3 ms whereas the model evaluation takes 4 ms. The time needed to retransform the coordinates is negligible.

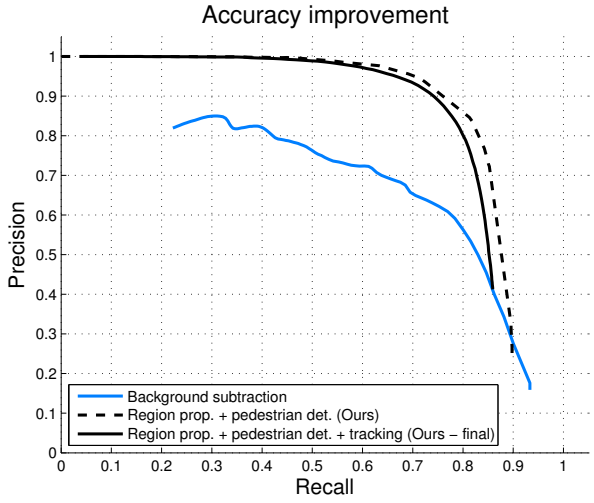


Figure 5.13: The obtained accuracy improvement compared to a naive background subtraction approach.

5.4.3 Comparative evaluation

Figure 5.13 illustrates the accuracy improvement we achieved as compared to a basic background subtraction technique, i.e. interpreting the foreground blobs that are large enough as pedestrians. As a score measure for each blob, the size of this blob is used. As seen, on these challenging images these naive methods yield poor results. The inclusion of our scene model and the application of a state-of-the-art pedestrian detector increases the accuracy enormously.

A quantitative comparison with other work using precision-recall curves on this dataset is difficult, since to the best of our knowledge no such accuracy results similar to our work exist. Existing work on these specific sequences of the CAVIAR dataset often focuses on activity recognition (e.g. *fight*) and anomaly detection. However, [93] present accuracy experiments using *tracking failure* measurements on 11 tracks of the CAVIAR dataset. For this, the authors consider a track lost if the tracking failed for 20 frames or more. In their work a multi-hypothesis tracking approach (particle filter) is used. Their achieved tracking failure versus the number of particles used is displayed in table 5.2. As seen, they achieve a tracking failure percentage of 33.6% with $N = 20$ particles and 16.8% when $N = 50$. Using our approach we achieve a tracking failure of 9.1% on the same sequences relying only on a single hypothesis tracker (Kalman filter).

Table 5.2: Overview of the tracking performance of [93] on the same dataset.

N (# particles)	**20**	**50**	**100**	**250**	**500**	**1000**
Track failure (%) of [93]	33.6	16.8	7.7	3.1	2.7	0

Figure 5.14: Qualitative comparison between running a detector on all scales and rotations (left) versus the output of our algorithm (right).

As a final qualitative analysis we compare our approach with a naive detection approach, that is running the standard deformable part model detector on all scales and all rotations. For this, we need to upscale the image five times (the smallest pedestrian to be detected is only 25 pixels high, and the height of the detection model equals 120 pixels), and use a rotation step size of 10 degrees. Using this approach, the calculation time for a single frame increases to about 13 minutes. Figure 5.14 displays the detections found using this naive approach (left), and the output of our algorithm (right). As seen, the naive approach yields several false positives and fails to detect all pedestrians. Our algorithm achieves excellent accuracy results with minimal computational cost (89 ms for this frame).

5.5 Conclusion

This chapter aimed to prove that our warping window approach, presented in the previous chapter, is easily generalisable to other scenarios with similar non-standard camera viewpoint as the blind spot camera application. For this, we applied this approach to the detection and tracking of pedestrians in challenging surveillance videos. In these surveillance scenarios a fixed camera is employed. As such, our proposed algorithm integrates foreground segmentation methods with scene constraints to generate region proposals, which are then

warped and evaluated by a single-scale pedestrian detector. As mentioned, using this approach we can employ a highly accurate pedestrian detector for non-trivial camera viewpoint images where existing pedestrian detectors fail, while still achieving real-time performance.

We performed extensive evaluation experiments concerning both accuracy and speed on the publicly available CAVIAR dataset. This dataset consists of typical low-resolution high-compression surveillance images taken with a wide-angle lens from a challenging viewpoint. We show that our approach achieves both excellent accuracy and processing speeds using a single-core CPU implementation only. Furthermore, our proposed method easily lends itself for a multi-threaded implementation.

In the next chapter we propose a methodology that allows for the combination of multiple pedestrian detectors in order to increase the detection accuracy.

Chapter 6

Combining pedestrian detectors to increase the accuracy

In chapter 4 we presented an initial approach that enabled the detection and tracking of pedestrians in the blind spot camera images. Although excellent results were achieved, further refinements are needed. The detection accuracy should further be increased, and to allow for the detection of e.g. bicyclists, the detection framework needs to be extended to multiclass detection. Therefore, in this chapter we present an approach that allows for an increase in accuracy. We propose a methodology to combine an arbitrary number of pedestrian detectors in order to increase the detection accuracy. This combination framework is a generalised method that is independent of the pedestrian detectors themselves.

For this, we exploit specific information from each pedestrian detector to determine the optimal combination parameters. Our main motivation for this approach is based on the fact that several pedestrian detection approaches are based on very different techniques (e.g. a different feature pool), and thus an efficient combination should yield higher accuracy results.

This methodology is easily integratable in the previous chapters, and as such allows for an increase in accuracy concerning the detection of pedestrians in the blind spot images. The work presented in this chapter is co-authored with F. De Smedt, and published at the ICPR 2014 conference [28]. We equally contributed to the development of the combination methodology and the implementation work.

6.1 Introduction

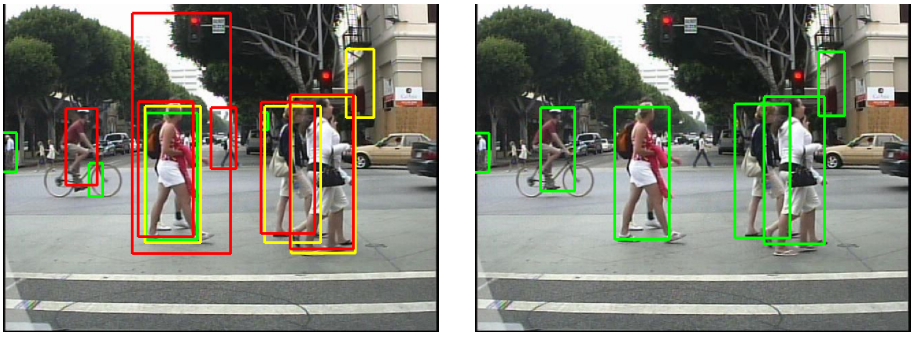
Pedestrian detection is an active research topic in recent years. Indeed, state-of-the-art algorithms achieve excellent accuracy results on challenging datasets (e.g. INRIA [21], Caltech [34]). As opposed to the optimisation of a single pedestrian detector (e.g. *Roerei* [6]), we propose a different approach to increase the detection accuracy: combining existing pedestrian detectors. Take for example the left frame of figure 6.1. Here, the detections of three different pedestrian detectors are visualised. Note that none of them manages to find all pedestrians, and all yield false detections. The optimal combination of all the different detection results we propose in this chapter yields the rightmost result, where all pedestrians are detected with minimal error. We therefore address a fundamental question: how should we combine the detection results of multiple pedestrian detectors to allow for a higher accuracy rate? Seeking such a strong combination rule is not a trivial task, since many design choices are to be considered in this process. This chapter tackles these questions, and presents a generic framework to achieve these pedestrian detector combinations.

Traditionally, combining multiple pedestrian detectors is performed using a basic *AND* or *OR* rule. Our framework utilises a more profound approach, and exploits information from each pedestrian detector. In a nutshell: our framework combines the detection scores from multiple pedestrian detectors using a weighted sum to calculate the final detection results scores. For that, we propose a measure for the complementarity between different detectors and the confidence of a detector in order to determine the optimal combination to further increase the accuracy. One could argue that to determine the optimal weights, a machine learning strategy (e.g. Support Vector Machines) could be utilised.

This is difficult since pedestrian detectors do not provide a detection probability score for all positions in the image. To achieve this, the pedestrian detection algorithms themselves should be adapted. However, since many pedestrian detection algorithms use completely different approaches, determining such a probability score in a fair way is an impossible task. Our approach allows to combine the detection results from all detectors *out-of-the-box* and, as we show further on, still manages to find the optimal combination weights. Note that our approach is applicable to arbitrary object detectors: here we use pedestrian detectors as an example application.

The main contributions presented in this chapter are two-fold:

- We propose a generic methodology that allows for an efficient combination of an arbitrary number of object detectors.



(a) Input image (b) Output of our approach

Figure 6.1: An example of how a combination of pedestrian detectors yields higher accuracy results. (Left) Red: LatV4-cc, Green: HOG, Yellow: ChnFtrs (Right) Green: output detections of our approach.

- We perform thorough experiments, and propose combinations that achieve better than state-of-the-art accuracy results.

Increasing the number of pedestrian detectors evidently increases the calculation time. A compromise needs to be determined between the accuracy and the computational complexity. This chapter focuses on accuracy improvement only. However, recent advances concerning speed improvements on existing pedestrian detectors show promising results.

Furthermore, fast multi-core implementations (GPGPU or multi-core CPUs) currently achieve reasonable to excellent processing speeds. Indeed, we demonstrated this in chapter 4, section 4.3 where we presented such a highly optimised hybrid CPU/GPU implementation of our VRU detection framework which achieved real-time processing speeds.

The remainder of this chapter is structured as follows. Section 6.2 describes related work on this topic. In section 6.3 we propose our combination approach and give detailed information on our combination parameters, followed by experimental results in section 6.4. Finally, in section 6.6 we present our conclusions.

6.2 Related work

Although integration strategies are applied in other research domains, to the best of our knowledge only few works concerning the optimal integration of multiple object detectors exist. Most work in which multiple object detectors are combined only use naive *AND* or *OR* combination rules. For example, [22] proposes the combination of a person model (DPM) and a face detector (VJ) to improve the detection results. A more intelligent approach is given in [66], where the authors present a probabilistic framework in which they combine the detection score of an object detector (DPM) with Conditional Random Fields (CRF) to achieve better scene understanding. However, they only use a single object detector. Recent work [72] uses multiple object models, based on the same detector (ChnFtrs), to cope with occlusion (new models are trained for a specific occlusion level). The different detection results are then in a weighted sum manner combined into a single detection score (with the area of each model taken into account).

In [125] the authors propose a detector ensemble based on false detections. Benenson et al. combines multiple pedestrian detectors, coined *Katamari* [7]. The work of Xu et al. [122] is closely related to our work, and was published only shortly after our framework. Here, the authors employed belief functions and evidential combination rules to combine different pedestrian detectors.

Our work significantly differs from all of the previously mentioned works. We propose an approach that aims to combine the detection results of multiple independent object (pedestrian) detectors in the most optimal manner. In essence, our goal is to determine the best possible combination rule to maximally increase the accuracy. We utilise specific information from each individual detector to obtain such an optimal combination rule. Furthermore our framework allows for the combination of an arbitrary number of detectors. As shown in section 6.4, the accuracy of an optimal combination outperforms existing state-of-the-art pedestrian detectors. In the next section we propose our approach and motivate our design choices.

6.3 Approach

As mentioned above, mostly only naive pedestrian detector combinations are used. In this work we try to increase the accuracy by combining the detection results of multiple pedestrian detectors in a more profound approach. That is, we propose to combine the detection scores from each individual pedestrian detector using a weighted sum. Our goal is then to find these optimal combination weights,

such that they exploit the strengths of each individual detector. For example, the DPM detector has excellent accuracy for high- to medium-resolution pedestrians, for which the accuracy of the HOG detector is lower; however small pedestrians can still be detected with HOG. Combining such detectors evidently yields better accuracy (as is done in [81]). The challenge now lies in the quantification of this information. Therefore, we propose the use of two different measures: *confidence* and *complementarity*.

- **Confidence**: The confidence value indicates how *good* a detector performs. It gives an indication about the probability of a detection by detector i being a correct detection (further indicated as $c_{\text{conf}(i)}$).
- **Complementarity**: Each pedestrian detector uses a specific design methodology (e.g. different feature pools or classifiers). This measure indicates how *different* the detectors are (further referred as $c_{\text{compl}(i)}$). As such, this measure captures the *added value* one detector gives over the other(s) when they detect a pedestrian at the same location.

We use these two measurements in a weighted sum to combine the detection scores from two or more pedestrian detectors (see subsection 6.3.3). Below we explain in detail how each of these measurements are determined.

Since each detector has specific tuning parameters (e.g. thresholds and non-maximum suppression) we need to determine an unbiased way to compute the confidence and complementarity coefficients over all detectors. To achieve this, we select a fair operating point (or detection threshold) on the Precision-Recall (PR) curve for each detector. We determined the optimal threshold using an *equal number of detection windows*.

Recall that, when varying the detection threshold each specific detector returns a different number of detections (i.e. bounding boxes). If a high threshold is used, only a limited number of detections (i.e. those with high confidence) are returned. If a low threshold is used, more detections will be returned. To allow for a fair comparison between all detectors we set their specific threshold such that all of them return an equal amount of detection windows. This equal amount of detection windows, further referred as N, is specified as a percentage of the number of ground truth detections.

This is visualised in figure 6.2 for three pedestrian detectors: *HOG* [21], *Integral Channel Features (ChnFtrs)* [32] and the cascaded *Deformable Part Models (LatV4-CC)* [43] where the blue points indicate the optimal threshold setpoints (N is chosen as 50% of the number of ground truth detections). In this chapter we use these three detectors as an example to illustrate our combination approach. An alternative method to retrieve the optimal operating points could be the use

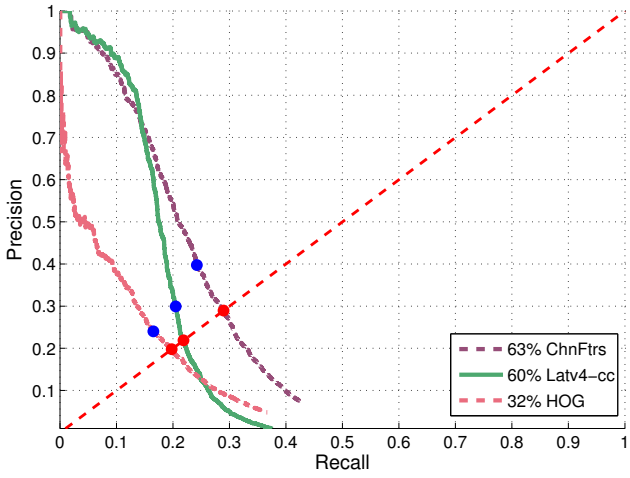


Figure 6.2: The optimal thresholds (PR operating points). Blue: obtained using a fixed number of detection windows ($N = 50\%$). Red: using an equal error rate line.

Table 6.1: Confidence coefficients for our three example detectors.

	HOG	**LatV4-CC**	**ChnFtrs**
Confidence coefficient	0.040	0.061	0.096

of a line determined by equal error rate. In this case, the intersection with this line and the PR curve gives the optimal thresholds (visualised by the dashed red line and dots).

In the next subsections we give an overview of how each of the coefficients defined above are determined. Note that all further calculations in the subsections below use this optimal threshold, thus ignoring detections with a lower detection score. We then explain how this information is used to combine the detection output of multiple detectors to achieve a higher accuracy. Finally, this section concludes with a validation of our approach, showing that it manages to reach the most optimal solution.

6.3.1 Confidence coefficient

The confidence coefficient $c_{\text{conf(i)}}$ gives an indication of the detection accuracy of detector i. This information is independent of other detectors in the combination

framework, and is based on the accuracy that a detector achieves on a specific (i.e. training) dataset. Recent, more accurate pedestrian detectors should evidently have a higher confidence coefficient. Several statistics can be used for this measure, e.g. the average precision (AP). We propose to use the area from the rectangle through the origin and the optimal operating point on the PR curve (indicated with the blue dots in figure 6.2) for each detector i.

$$c_{\mathrm{conf}(i)} = P_{opt(i)} R_{opt(i)} \tag{6.1}$$

Thus, these confidence coefficients are calculated beforehand (i.e. in an offline stage) based on the detection results of these detectors on a training set. Table 6.1 gives the confidence coefficients for our example detectors.

6.3.2 Complementarity coefficient

Different detectors use different design methodologies and feature pools. Therefore, each pedestrian detector reacts differently to a specific image patch. Combining pedestrian detectors that are complementary with respect to each other thus could yield better detection results. Our complementarity coefficient $c_{\mathrm{compl(i)}}$ tries to indicate how *different* these pedestrian detectors react, and thus how complementary they are. When multiple detectors with very different detection approaches yield a detection at the same image location, the chance of that being a true detection increases significantly (much more than when e.g. multiple detections from rather redundant pedestrian detectors using the same approach are found).

As an example, the frame in figure 6.3 visualises three detector outputs (see caption for colour coding). These detectors give significantly different detections. For example, only HOG manages to find the small pedestrian on the left, and some detectors generate different false positives. Some locations are covered by more than one detector, indicating a higher probability that these are correct detections. If combined efficiently, an optimal accuracy is achieved. To determine a complementarity coefficient for each detector, we first calculate the pairwise complementarity score $w_{i,j}$ between two detectors i and j.

This is done as follows. We first compare the detection performance on a training dataset. Here, the Caltech training set is used (as will be discussed in section 6.4). For each frame, each detector is compared in a pairwise manner. The number of detections from a specific detector which are not covered by the other detector (using the 50% intersection over union criterion of Dollár [34]) is determined. These are then summed over all frames, and divided by the total number of detections for that detector. For example (on one frame), in figure 6.3 *ChnFtrs* has four detections of which three are covered by *LatV4-CC*,

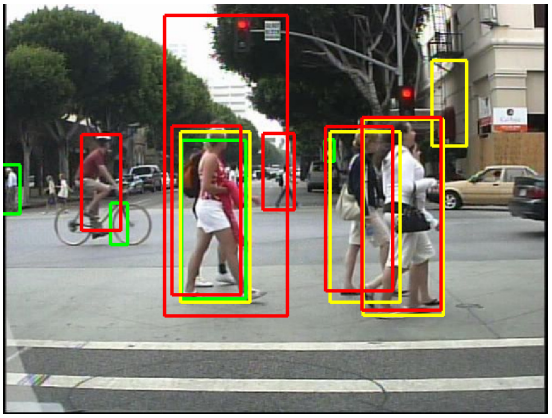


Figure 6.3: Example detections on Caltech frame. Red: LatV4-CC, Green: HOG, Yellow: ChnFtrs.

thus $w_{ChnFtrs,LatV4-CC} = 25\%$. *HOG* covers one detection from *ChnFtrs* resulting in $w_{ChnFtrs,HOG} = 75\%$. If done for each detector pair, this results in a square complementarity matrix, visualised for seven detectors in figure 6.4. Thus, similar to the confidence coefficients, this complementarity matrix is calculated beforehand.

Note that no annotation data is used; the fact that a detection is correct or not is irrelevant for the complementarity coefficient. This information is already included in our confidence coefficient. The complementarity coefficients aims to indicate how much extra information (i.e. added value) each specific detector introduces in the case of overlapping detections. Several interesting conclusions are retrieved from figure 6.4. Note that a high value in this matrix indicates much added value. As seen, all scores for the combinations of *ChnFtrs*, *FPDW* and *ACF* are low. This is evident, since all three detectors come from the same family and employ the same features.

For example the *VJ* detector is highly complementary to the other detectors. However, this is due to the fact that this detector returns a significant amount of false positives, and thus yields several bounding boxes which are not found by the other detectors. Thus, an optimal combination is found when combining detectors which have both a high confidence and complementarity with respect to each other.

During the effective (online) combination of the detection results on a new dataset, this pre-computed complementarity matrix is used to calculate the complementarity coefficients of overlapping detections in this test set as

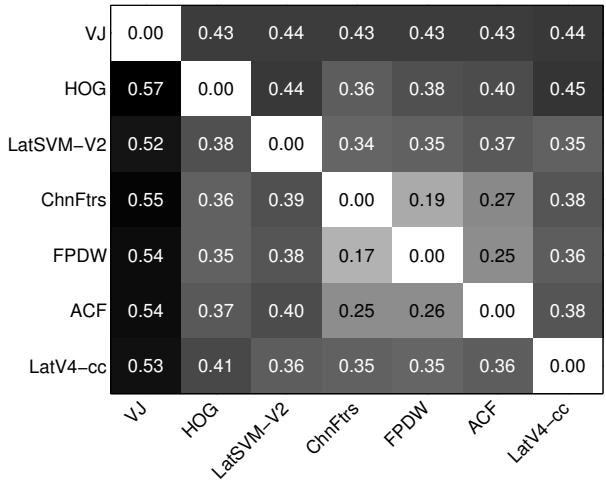


Figure 6.4: The complementarity matrix visualised for seven detectors.

follows. For overlapping detections, we first extract the corresponding square submatrix (containing only the *relevant* detectors - those that account for one of the detections in the overlap) from the total complementarity matrix. Next we calculate a single *average complementarity coefficient* C_i for each detector i involved in this overlapping detection, using the individual pairwise complementarity scores $w_{i,j}$:

$$C_i = \frac{\sum_{j=1}^{n} w_{i,j}}{n-1} \tag{6.2}$$

Where n indicates the number of relevant detectors involved in the overlap. Intuitively, if $C_i = 0$ this detector is totally redundant and does not yield extra information, if $C_i = 1$ it is perfectly complementary with respect to the other relevant detectors. In a final step we use these averaged complementarity scores C_i from all individual relevant detectors to determine the final complementarity coefficient $c_{compl(i)}$ for a single detector i as follows.

As a simplified case, take for example a combination of two detectors A and B which have overlapping detections, thus $n = 2$. If one of both detectors is completely complementary ($C_i = 1$) it yields valuable information and thus we set $c_{\mathrm{compl}(i)} = 1$, independent of the other average complementarity coefficient. If both are equally complementary ($C_A = C_B$) they are given the same score $c_{\mathrm{compl}(A)} = c_{\mathrm{compl}(B)}$. In the extreme case where both are completely redundant ($C_A = 0, C_B = 0$), this score equals $c_{\mathrm{compl}(A)} = c_{\mathrm{compl}(B)} = 1/2$. Intermediate

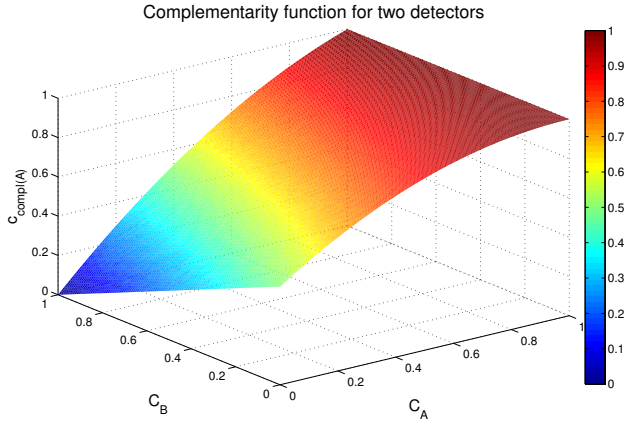


Figure 6.5: The complementarity function $c_{\text{compl}(A)}$ for two detectors.

values are calculated based on both average complementarity coefficients. For one of both complementarity coefficients (A) this yields:

$$c_{\text{compl}(A)} = \frac{1}{2}\left[1 + (C_A - C_B) + C_A C_B + C_A(1 - C_A)\right] \tag{6.3}$$

For clarification, the complementarity function $c_{\text{compl}(A)}$ for $n = 2$ is visualised in figure 6.5. For more than two detectors, this complementarity function generalises as follows:

$$\begin{aligned} c_{\text{compl}(i)} = \frac{1}{n}\Bigg[1 + \sum_{\substack{j=1 \\ j\neq i}}^{n}(C_i - C_j) + \sum_{\substack{j=1 \\ j\neq i}}^{n}(C_i C_j) \\ +(\sum_{j=1}^{n} C_j - 1)(1 - C_i) + \prod_{j=1}^{n}(1 - C_j)\Bigg] \end{aligned} \tag{6.4}$$

Where C_i are the individual average complementarity coefficients and n indicates the number of relevant detectors involved in the overlap. These n complementarity coefficients thus summarise how each individual detector involved in this overlap should be weighted as such to maximally exploit the information potential of each specific detector.

To illustrate, suppose that an overlapping detection with both *HOG* and *ChnFtrs* is found ($n = 2$). First, we extract the 2×2 complementarity submatrix.

Next, using equation 6.2, we determine the average complementarity coefficient for each detector ($C_{HOG} = 0.45, C_{ChnFtrs} = 0.36$). Finally, we calculate the complementarity coefficient for each detector using equation 6.3, yielding $c_{compl(HOG)} = 0.685$ and $c_{compl(ChnFtrs)} = 0.675$.

6.3.3 Combining detection results

Using the coefficients determined above, the actual combination on a new dataset is performed as follows. On each input image of this dataset, all detectors are run separately. A first step consists of the normalisation of all detection scores, since the scores significantly differ for each detector. This is simply achieved using the standard score approach (subtract average and divide by standard deviation). Then an offset is added (the minimal score over all detections) to ensure that all detection scores are positive. Next, we determine where and which pedestrian detectors have an overlapping detection (using the 50% intersection criterion). In this case a final detection score is determined based on the output scores S_i of the n overlapping detections, the confidence coefficient and the complementarity coefficient of each detector i that yielded a detection there, calculated as mentioned above, using a weighted sum:

$$S_{\text{final}} = \sum_{i=1}^{n} c_{\text{conf}(i)} c_{\text{compl}(i)} S_i \tag{6.5}$$

In [24], De Smedt further refined this approach by considering the clustering of the detections as a graph problem. There, the clustering problem was reduced to finding so-called *cliques* in graphs. For this, the algorithm introduced in [13] is used. This approach further increased the accuracy of our combination methodology. However, in this chapter we present the accuracy results as given in our original publication [28].

For the final bounding box we return the weighted average (using the individual detection scores) over the overlapping bounding boxes. For all non-overlapping detections we multiply the detection score with the confidence value of that detector and the complementarity coefficient, calculated as if this detection overlapped with all detectors. Our confidence and complementarity coefficients are chosen in such a way that multiple detections from complementary detectors with high confidence return high scores, whereas redundant detectors with low confidence evidently output lower total detection scores. In section 6.4 our experiments show that our approach achieves very good accuracy results.

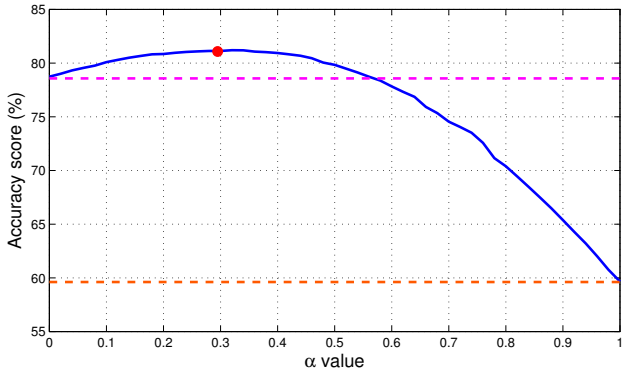


Figure 6.6: The accuracy score (%, blue curve) in function of the α value. Dashed lines indicate the accuracy of both individual detectors (*HOG*: Orange, *ChnFtrs*: Magenta). The red dot is the combination rule we propose, derived independently from this experiment.

6.3.4 Validation of our approach

In this section we validate our combination rule, and show that our weighted sum approach using the confidence coefficient and complementarity coefficient as defined above reaches the most optimal solution (i.e. the combination with the highest accuracy). This is done as follows. Take for example the combination of two detectors A and B. Our combination rule then becomes:

$$S_{\text{final}} = c_{\text{conf}(A)} c_{\text{compl}(A)} S_A + c_{\text{conf}(B)} c_{\text{compl}(B)} S_B \tag{6.6}$$

Since only the relative detection scores are important, this can be reformulated as:

$$S'_{\text{final}} = \alpha S_A + (1 - \alpha) S_B \tag{6.7}$$

Where:

$$\alpha = \frac{c_{\text{conf}(A)} c_{\text{compl}(A)}}{c_{\text{conf}(A)} c_{\text{compl}(A)} + c_{\text{conf}(B)} c_{\text{compl}(B)}} \tag{6.8}$$

Thus, if we let α vary from zero to one, all possible relative combinations are evaluated.

We calculated for each of these combinations the accuracy score of the resulting combined detector on the test set. Figure 6.6 gives these results for the combination of *HOG* and *ChnFtrs*. At the extreme values 0 and 1 of α, the accuracy score equals that of both individual detectors (indicated with the dashed lines). An optimal combined accuracy is reached for a specific value of α between these two boundaries. The red dot indicates the value of α calculated

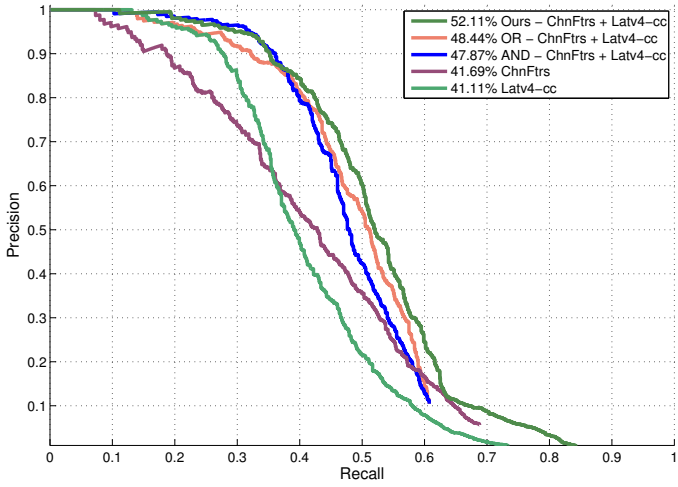


Figure 6.7: Precision-Recall curve of our combination approach for *ChnFtrs + LatV4-cc*, compared with the standard *AND* and *OR* combinations.

using our combination rule (with $c_{\text{compl}(HOG)} = 0.685$, $c_{\text{compl}(ChnFtrs)} = 0.675$). As can be seen, our proposed combination rule manages to find the most optimal combination weights. Note that our weights are calculated based only on the confidence and complementarity measures. These are easily extracted from the detection results, thus avoiding the need to perform an exhaustive search over all possible combinations like we do in this validation experiment.

6.4 Experiments and results

To illustrate the potential of our combination approach, we performed thorough accuracy experiments. Our framework uses the publicly available detection results from the Caltech dataset [34]. This dataset consists of about 250,000 frames of which each 30th frame is used for evaluation (resulting in about 8300 frames).

All the needed combination data (i.e. the confidence coefficients and the complementarity matrix) were first calculated on the training set (set00 - set05, 4250 frames). As optimal operating point we use $N = 50\%$ of the number of ground truth detections. Next, our combination rule was executed on the test set (set06 - set10, 4024 frames), using the *reasonable* settings (see chapter 3, section 3.4 for more information on this specific setting). The experiments

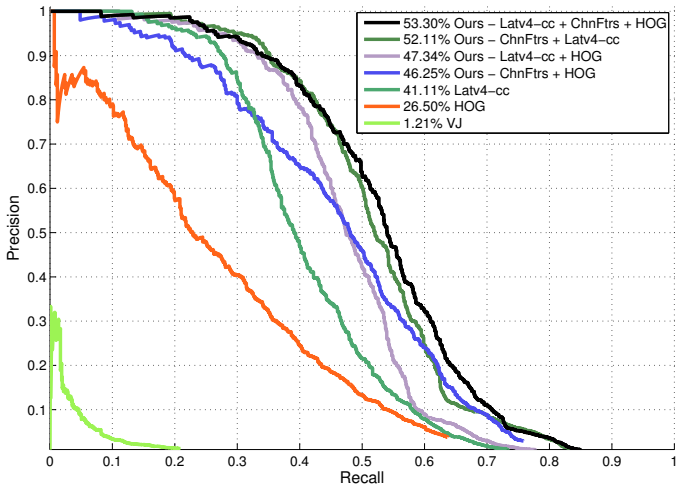


Figure 6.8: Precision-Recall curve of all possible combinations of our three benchmark detectors. As seen, combining all three detectors yields the highest accuracy.

indicate that our combination approach achieves excellent accuracy results, and specific combinations achieve better than state-of-the-art detection results.

Besides our approach, for each experiment we also performed the *AND* (only keep overlapping detections) and *OR* (keep all detections) combinations. Figure 6.7 displays the precision-recall results of our combination rule for *ChnFtrs + LatV4-cc*, compared with these AND and OR results. Our proposed combination approach easily outperforms these naive combination rules. Further note that the recall of our approach is significantly higher. A similar trend is noticed for our other combinations versus the AND and OR combination rules.

Figure 6.8 compares all possible combinations of the three benchmark detectors used in this chapter. A combination of two detectors evidently outperforms both corresponding individual detectors. A combination of the three detectors further slightly increases the accuracy.

Finally, figure 6.9 displays the accuracy of our combination rule for the combination of our three benchmark detectors, compared with the current state-of-the-art pedestrian detectors. As can be seen our combination rule achieves excellent detection results, reaching an accuracy of 85.32% on the challenging Caltech dataset.

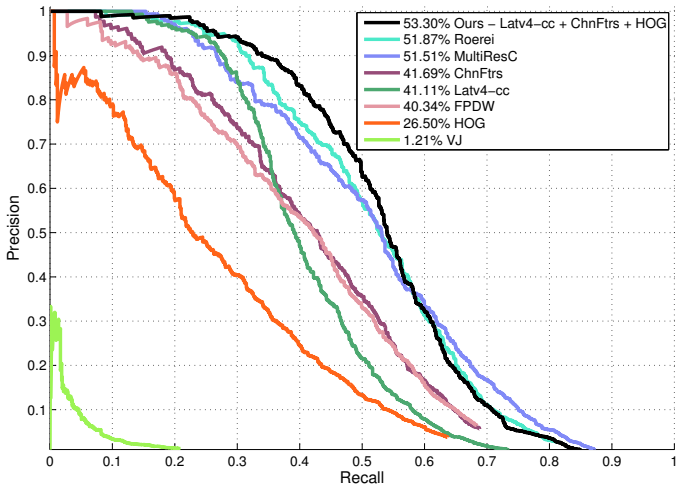


Figure 6.9: Accuracy of our combination rule for the three benchmark detectors versus the state-of-the-art pedestrian detectors (*Roerei* [6], *MultiResC* [81], *ChnFtrs* [32] and *FPDW* [31]). Our combination methodology achieves state-of-the-art accuracy results.

6.5 Discussion

We would like to emphasise that the combination methodology proposed in this chapter was developed in an empirical manner. The main goal of this chapter was to develop an efficient method in which the final bounding boxes of different object detectors could be combined to minimise the number of false positive detections while maximally maintaining true positive detections. For this, we presented measures to capture both how good the individual detectors are and how much added value specific detectors give over other detectors. These measures are then used in a weighted sum to define a new score to each bounding box. Note that a more profound (e.g. more mathematically founded) combination methodology might exist. However, our experiments indicated that using our empirically determined methodology, excellent accuracy results were obtained.

6.6 Conclusion

In this chapter we presented a generic pedestrian detector combination methodology to further increase the detection accuracy of our automatic blind spot guarding system. Our approach allows for the combination of an arbitrary number of pedestrian detectors, and manages to achieve an optimal combination rule. Our framework is independent from the specific object that ought to be detected, here we employ pedestrian detection as an example.

To achieve this optimal combination, we introduced two measures: confidence and complementarity. Using these measures the detection scores of multiple pedestrian detectors are combined in a final detection score. We validate that our approach is indeed able to find the most optimal combination. Our experiments, using three standard pedestrian detectors, indicate that we outperform the traditionally used (naive) *AND* and *OR* approaches, and achieve better than state-of-the-art detection results on the challenging Caltech dataset.

In the next chapter we refine our warping window approach presented in chapter 4 in two ways. We introduce an approach to increase the detection accuracy and evaluate if a combination of multiple pedestrians detectors could further increase the accuracy. Furthermore, we extend our framework to cope with multiclass object detection.

Chapter 7

Extending the VRU detection system

In chapter 4 we presented a methodology that enables the detection of pedestrians in the blind spot images. As mentioned, this was only an initial approach: further refinements to increase the detection accuracy and allow for multiclass detection are needed. In the previous chapter we presented a methodology that, using a combination of multiple pedestrian detectors, allows for an increase in accuracy. In this chapter we now present two distinctive extensions of our initial *warping window* approach presented in chapter 4.

First, in section 7.1 we discuss two different approaches to further increase the detection accuracy. We increase the detection accuracy by taking into account the perspective distortion induced by the blind spot camera. Furthermore, we perform experiments to evaluate if the detection accuracy could further be increased when using multiple pedestrian detectors (similar to chapter 6). This work was published at the ICPRAM 2014 [106] conference and an extended version was published as a book chapter in LNCS in 2015 [108].

Next, in section 7.2 we extend our approach to other road users than pedestrians. As discussed in chapter 2, apart from pedestrians, often also bicyclists are the victim of these blind spot accidents. For this, we propose a methodology that extends our framework to include bicyclists in an efficient manner. We published this work at the ACPR 2015 conference [107].

Figure 7.1: Similarity versus perspective transformation model.

7.1 Increasing the accuracy using a perspective warping window approach

7.1.1 Introduction

In chapter 4 we proposed our pedestrian tracking framework for genuine blind spot camera images. For this, we proposed the use of our warping window approach. However, this initial approach was based solely on a naive similarity warp, running up against its limit (e.g. with respect to the accuracy for our application). In this section we propose a multi-pedestrian detection and tracking framework based on our *perspective warping window approach*: we extensively redesigned and improved our previous work making it more elegant and accurate, without significantly increasing the algorithmic complexity. On the contrary, we even obtain higher computation speeds. Figure 7.1 concisely compares our previous and our improved novel approach presented here.

The framework presented in this section is similar to that proposed in chapter 4. As a reminder, the algorithm proposed there briefly works as follows. Traditional state-of-the-art pedestrian detectors use a sliding window paradigm: each possible position and scale in the image is evaluated. These standard pedestrian detectors are not directly applicable on our blind spot camera images. Due to the specific viewpoint, pedestrians appear rotated and scaled. Evaluating all positions, rotations and scales is not feasible for real-time applications. Instead, we eliminate the need to perform such full scale, rotation and position search using the exploitation of scene constraints. That is, at each position in the input image we locally model the transformation induced by the specific camera viewpoint and the lens distortion.

The main novelty proposed in this section is found in the way this exact

transformation is modelled. In the previous chapters only a naive rotation and scaling is used: here we extend our approach and employ a perspective transformation resulting in a more accurate and even faster framework. The remainder of the detection pipeline is identical: during detection, we warp the regions of interest (ROIs) in the image and use a standard pedestrian detector at a single scale on each ROI. This approach is again integrated in a tracking-by-detection framework and combined with temporal information, making it more robust while reducing the detection time.

Cho et al. [17] proposed a pedestrian tracking framework related to our work, exploiting scene constraints to achieve real-time detection. However, they use a basic ground plane assumption whereas our approach is much more flexible and generic. Moreover, our specific datasets are much more challenging due to the severe distortion. We thus significantly differ from existing approaches. We aim to develop a monocular multi-pedestrian tracking framework with a challenging backwards/sideways looking view, targeting high accuracy at real-time performance.

Furthermore, most of the classic sliding window approaches assume only object scale variation. Other geometrical variations (e.g. rotation [60] and aspect ratio [73]) are usually covered by an exhaustive search approach. Our proposed warping approach offers a solution that can even cope with perspective distortion. In fact, without our warping window paradigm it would be unfeasible in practice to perform such an exhaustive search in a perspective distortion space.

The remainder of this section is structured as follows. First, in subsection 7.1.2 we discuss the extension of our warping window approach. In subsection 7.1.3 we then discuss which pedestrian detector is most suited to be utilised in our framework. In the previous chapters we utilised the cascaded deformable part model in our detection pipeline. Here, we evaluate an additional pedestrian detector, and present experiments in which we evaluate if a combination of two pedestrian detectors could further increase the detection accuracy. In subsection 7.1.4 we discuss the integration of this approach in a tracking-by-detection framework.

As in the previous chapters, we performed extensive experiments to evaluate our algorithm concerning both speed and accuracy. These are discussed in subsection 7.1.5. For this, we recorded additional realistically simulated dangerous blind spot scenarios with a real truck and thus extended our own recorded dataset that was used to evaluate our initial warping window approach proposed in chapter 4.

7.1.2 Extended warping window approach

The main detection methodology remains identical to our previously introduced approach: given our standard blind spot camera images, pedestrians appear rotated, scaled and distorted. If we assume a flat ground plane, these transformation parameters only depend on the specific position in the image. If we know the transformation we can model the distortion for that ROI, extract and warp the ROI image patch to a fixed scale and perform pedestrian detection on a single scale only.

An extension of our warping window approach is found in a refinement of the exact warping transformation that is employed. In chapter 4 we modelled this transformation as a simple *similarity* transform consisting of only a scaling and rotation of the image patch. However, in practice this is not the case, and a similarity transformation is in fact only an approximation of the exact transformation. In this section we further refine the transformation step, and model the transformation as a perspective distortion. For this, we calibrate our blind spot camera and use the vantage point in the undistorted image domain to determine the perspective model, as seen in the right image of figure 7.1.

Figure 7.2 illustrates our perspective warping window approach. Apart from how the distortion is locally modelled (the first few steps in this figure), the detection pipeline is identical to our previously proposed warping window methodology. We thus rewarp the image patches to fixed and upright pedestrians and employ a one-scale only pedestrian detector. We thus eliminate the need to construct a scale-space pyramid. Note that although we perform detection on a single scale only, the pedestrian model still provides some invariance with respect to the pedestrian height due to the flexible deformable parts model.

However, if large deviations from the standard height (e.g. children) need to be detected, an extra scale needs to be evaluated. This is demonstrated in the next chapter, where we further refine our framework to also detect children. After detection, the resulting bounding boxes are then retransformed to the input image domain, and integrated into our tracking-by-detection framework. We now describe further details of our algorithm: how this position-specific transformation is mathematically modelled and how the calibration is performed.

Figure 7.3 illustrates how the transformation is locally modelled. We use a perspective distortion model in the lens-distortion-corrected image. At each position, the height and width (at the ground) are known after a one-time calibration step (see further). These are visualised as two heat maps (the so-called look-up-functions or LUFs) in figure 7.3. The transformation coordinates are determined as follows. Each ROI centre coordinate (indicated with the red asterisk in the leftmost image) is first transformed to the undistorted image.

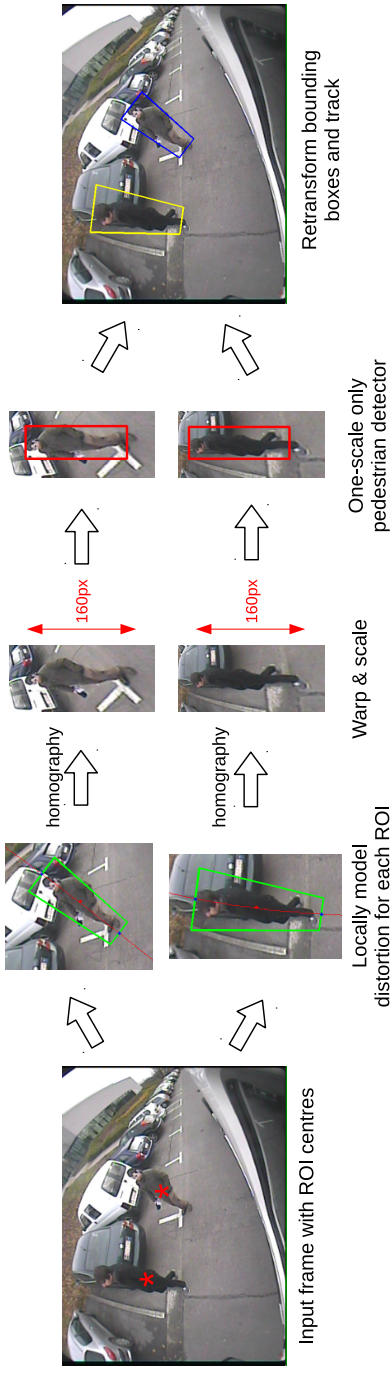


Figure 7.2: Illustration of our novel perspective warping window approach. At each position in the image we locally model the distortion, warp the ROIs to a standard scale and use a one-scale only pedestrian detector. As compared to figure 4.6, here the local distortion is modelled as a perspective transformation.

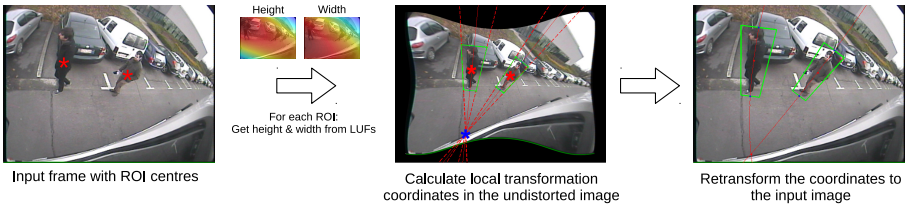


Figure 7.3: The transformation is modelled as a perspective transformation, calculated in the undistorted image.

This lens undistortion is simply based on the traditionally used radial lens distortion model:

$$\mathbf{x}' = \mathbf{x}(1 + k_1 r^2 + k_2 r^4) \tag{7.1}$$

$$r^2 = x^2 + y^2 \tag{7.2}$$

Here, $\mathbf{x}'$ denotes the corrected pixel coordinate, $\mathbf{x}$ the input coordinate and k_1 and k_2 indicate the radial distortion coefficients. Next we construct the vertical vantage line through this ROI centre in the undistorted image, and retrieve the height and width (at the bottom) from the two calibration LUFs. Indeed, due to the slightly downward-looking camera viewpoint all vertical lines in the world intersect in the image in a vantage point which is positioned directly below the blind spot camera (visualised in the middle of figure 7.3 as the blue asterisk). We perform camera calibration and undistort the image to rectify these vantage lines.

Based on these data we construct the perspective model in the undistorted image. The rotation of the image patch is determined from the angle of the vantage line, and the length ratio between the top and bottom is calculated based on the distance to the vantage point. We thus locally model the pedestrians as if they are planar objects standing upright, faced towards the camera (that is, perpendicular to the optical axis of our blind spot camera). Our experiments show that this is a valid approximation for pedestrians.

These coordinates are then retransformed to the distorted input image. Note that evidently only the coordinates are transformed, the middle image displayed here is only used for visualisation purposes. Based on the coordinates in the distorted image, and the known calibration data we apply a homography on the ROI image patch, thereby effectively undoing the local perspective distortion (visualised in figure 7.2).

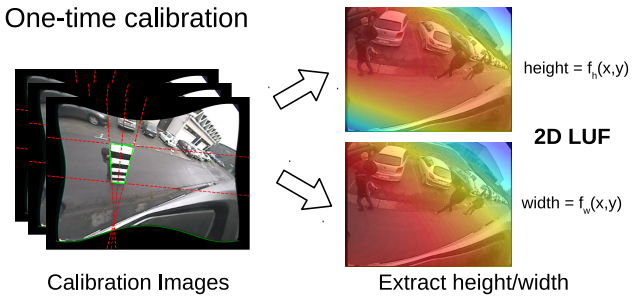


Figure 7.4: A one-time calibration is needed to determine the local perspective distortion.

To obtain these two LUFs, a one-time calibration step is needed. To achieve this, we manually annotated about 200 calibration images. We utilised a planar calibration board of 0.5×1.80m, and captured calibration positions homogeneously spread over the entire image (figure 7.4). The labelling was performed in the undistorted image. These images yield the vantage point, and the height and width of a pedestrian (at the ground) at each position for that image. Next we interpolated these data points using two-dimensional second order polynomial functions for both the height and the width: $f_h(x, y)$ and $f_w(x, y)$ with:

$$f_i(x, y) = p_0 + p_1 x + p_2 y + p_3 x^2 + p_4 xy + p_5 y^2 \tag{7.3}$$

Both functions are displayed as heat maps in figure 7.4: for each pixel coordinate they effectively give the height and width of the calibration board at that location. If for some reason the position of the camera with respect to the ground plane changes, a recalibration needs to be performed. This is highly unlikely though, due to the robust camera mounting on the truck.

Thus to summarise, detection is composed of four different steps: calculate the local perspective distortion model at each ROI centre, perform a homography and transform the pedestrians to an undistorted, upright position at a fixed height, run a pedestrian detector at one scale, and finally retransform the coordinates of the detected bounding boxes to the original input image. In the next section we discuss which pedestrian detector is most suited, and if a combination of multiple pedestrian detectors could further increase the detection accuracy.

7.1.3 Evaluating pedestrian detectors for our framework

In previous chapters we utilised the deformable part-based detector as pedestrian detector in our frameworks. In this section we evaluate the performance of an additional pedestrian detector based on a rigid detection model. Furthermore we investigate if a combination of multiple pedestrian detectors could further increase the accuracy.

Rigid or part-based model

Based on the comparative results discussed in chapter 3 we conclude that, since we aim for high accuracy, two approaches towards pedestrian detection are most suited for our application: the deformable part-based HOG models (DPM) [43], and the rigid model approaches such as the *Fastest Pedestrian Detector in the West* (FPDW) [31]. The FPDW has only slightly lower accuracy on established datasets [35] and is much faster. However, since we need to evaluate only one scale, no feature pyramid is constructed, thus this speed advantage is here not relevant. Selecting the most appropriate pedestrian detector for integration in our framework thus boils down to the selection of the most accurate detector on our dataset. For this we performed accuracy measurements for both detectors. To perform a fair comparison, let us briefly summarise how both pedestrian detectors work.

As mentioned in chapter 3, the DPM detector uses a pre-trained model, consisting of HOG features. It consists of a root filter and a number of part filters representing the head and limbs of the pedestrian. To use this model, first a scale-space pyramid is constructed, using repeated smoothing and subsampling. For each pyramid layer the HOG features are computed. Then, for a specific scale the response of the root filter and the feature map is combined with the response of the part filters to calculate a final detection score. Recall that on our 640×480 resolution images this detector off-the-shelf needs an average of 2.8 s per frame, while their cascaded version needs on average 640 ms per frame.

As opposed to the deformable part-based detector, the FPDW detector utilises a rigid model, not making use of deformable parts. Again, a scale-space pyramid is constructed. Next, features are calculated on transformed versions of the original input image, called *channels*. Examples are colour channels and gradient channels. The most basic features are a sum over a rectangular region in one of the channels. These features are combined into higher-order features, and classification is performed using a depth two decision tree boosted classifier (AdaBoost). Fast rejection of candidate windows is possible through the use of a *soft-cascade* approach. Essentially, FPDW uses ICF as a baseline, and

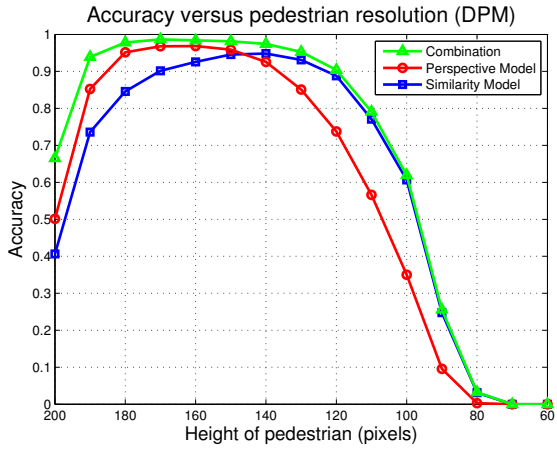


Figure 7.5: Determining the optimal scale parameter. Results for the deformable-part detector.

achieves a speedup through a more efficient construction of this feature pyramid; intermediate feature scales are approximated using scales nearby, avoiding the need to compute all feature scales. Out-of-the-box calculation time for ICF on average equals 451 ms per frame (on 640×480 images), while using the FPDW approach the calculation time drops to 147 ms per frame.

We altered both detectors into single-scale detectors to integrate them in our framework. Next, we need to determine the most optimal scale for each detector; the rescale height at which the maximal accuracy is reached on our specific dataset. For the DPM model such experiments were already performed in chapter 4. However, since the transformation of the patches is altered (using the perspective approach), the optimal scale might differ from the previously determined 140 pixels.

Determining the optimal scale

As mentioned, we rescale the pedestrians to a fixed height in order to reduce the calculation time. For this, an optimal value needs to be determined. To achieve this, we labelled and extracted 6000 pedestrian images at different locations in the images from our dataset, and performed the warp operation as given above. These pedestrians were warped to fixed heights, and we then performed accuracy measurements with both single-scale pedestrian detectors to determine the optimal height for each of them. As in chapter 4, subsection 4.2.2, here the accuracy is also defined as the *true positive rate* ($TPR = \frac{TP}{TP+FN}$).

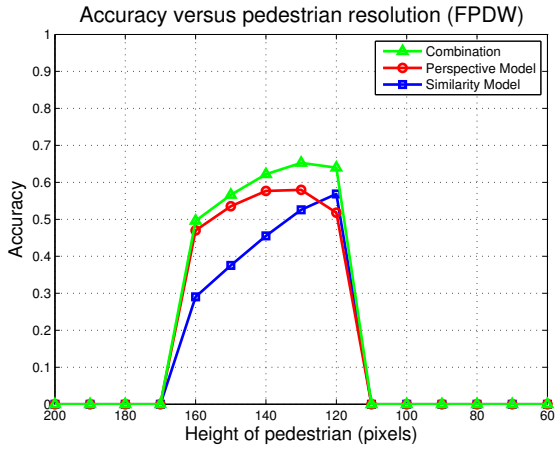


Figure 7.6: Determining the optimal scale parameter. Results for the FPDW detector.

Besides our *perspective transformation model* presented in this chapter, we also warped the pedestrians using the *similarity transformation model* as presented in chapter 4, simply consisting of a rotation and scaling operation (see figure 7.1 for a qualitative comparison). This was done to analyse the benefit of our more complex perspective model. Figure 7.5 and figure 7.6 display our results for both pedestrian detectors. Besides the individual transformations, we also give the combined accuracy (obtained using a standard OR operation).

Evidently, optimal accuracy is reached when pedestrians in the rescaled image patches approximate the height of the detection models. Note that the one-scale DPM detector achieves much better accuracy results for all transformation models as compared to the FPDW detector. The reason for this significant difference is found in the design methodology of both detectors. Due to the part-based approach, the DPM detection model is much more flexible, making this detector invariant to slight deviations in height between the pedestrians that need to be detected, and the actual pedestrian model.

The FPDW is much more sensitive for this due to the rigidness of the detection model. Since in our image patches slight differences between the actual and estimated pedestrian height exist (due to small calibration errors and the inherent height differences between pedestrians), more search scales would be needed to obtain higher accuracy with the FPDW approach. Further note that for both detectors the optimal resolution of the perspective and similarity transformation model differs. Concerning DPM, for the perspective transformation model the optimal height lies at 160 pixels, whereas the similarity

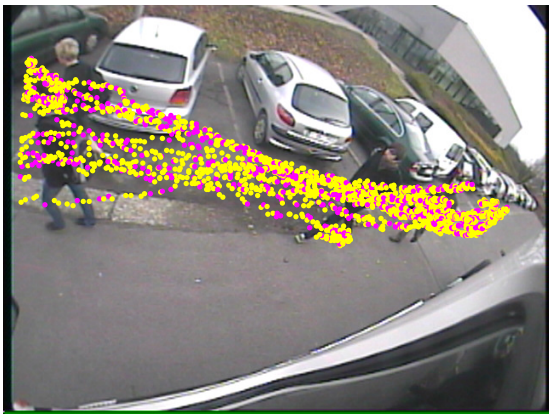


Figure 7.7: Performance of both pedestrian detectors in function of the position in the image. Coloured dots indicate which pedestrian detector performed best. Yellow: DPM. Magenta: FPDW.

model reaches its optimum at 140 pixels. As can be seen, the perspective model has a clear accuracy advantage over the similarity model. If both models were combined, an even higher accuracy is achieved. This, however, would increase the calculation time. Although at lower accuracy, a similar trend is noticeable for the FPDW detector.

Position dependence

These insights favour the use of the deformable part-based model over the rigid FPDW approach. However, these experiments exclude the influence of the position in the image with respect to the detection accuracy. Specifically for our images, the position inherently defines the scale and the amount of transformation. For example, specific positions in the image require significant upscaling to transform these patches to the fixed height. Furthermore, research indicates that rigid models perform better on low-resolution pedestrian patches, whereas part-based models achieve higher accuracy on large pedestrian patches [81]. Thus, a combined approach – where the *best* detector is selected using the spatial location in the image – may increase the accuracy.

We performed such experiments to evaluate this hypothesis as follows. The *best* detector is defined as having the highest detection score on the same image patch. Since the range of detection scores for both pedestrian detectors differs, first a normalisation of the detection scores is applied. For this, we warped the 6000

Figure 7.8: Performance of the two transformation models in function of the position. Coloured dots indicate which model performed the detection. Red: perspective model. Blue: similarity model. Green: both models. Yellow indicates missed detections.

pedestrian image patches mentioned above using the perspective transformation to the optimal height of 160 pixels. These patches are then equally divided in a *training* and a *test set*. Next, both pedestrian detectors were evaluated on the test set, and their detection scores were normalised (subtract the average and divide by standard deviation – determined on the training set). This allows for a fair comparison between both detectors. For each patch, the detector with the highest detection score (if both found a detection) is assigned as optimal detector for this patch. These results are visualised in figure 7.7 in function of the position in the image. The coloured dots indicate where which pedestrian detector performed best (yellow: DPM, magenta: FPDW). As visualised, no specific image location is favoured by any detector. Therefore, currently we do not perform such a combination, and utilise a single pedestrian detector.

Evaluation of the chosen detector

Based on all experimental results mentioned above, we thus opted for the cascaded deformable part-based models as baseline detector in our framework: it achieves excellent accuracy results, lends itself perfect to perform *true* single scale detection and, due to this single scale approach, achieves excellent speed results (as shown further). Figure 7.8 shows where each transformation model performs best in function of the position in the image (only for the deformable part-based model). Red dots indicate where the perspective model performed

Figure 7.9: Example of five initial search coordinates together with their corresponding transformation ROIs.

best, blue where the similarity model performed best and green were both models found the detection. Yellow indicates a missed detection. The perspective model obviously performs much better than the similarity model. The similarity model performs slightly better only at the image border, due to the small calibration error there. The perspective model performs better close to the truck because of the large amount of viewpoint distortion there. Note that if we analyse positions where both models found the pedestrian, the perspective model achieves the best detection score in 69% of these cases, further indicating its clear advantage over the similarity transformation model.

7.1.4 Tracking framework

To further improve the accuracy and detection speed we integrated our warping window approach in a tracking-by-detection framework. Instead of a full frame search, we use initial search coordinates (which define transformation ROIs) at the border of the image, and initially only perform detection there. See figure 7.9 for an example of such initial search coordinates.

This tracking-by-detection frame is again identical to the tracking framework in which our initial warping window approach was integrated (see chapter 4, subsection 4.2.2). We employ a linear Kalman filter with constant velocity motion model. However, as an additional experiment to increase the accuracy we included a scale factor in the state vector. This state vector x_k then becomes:

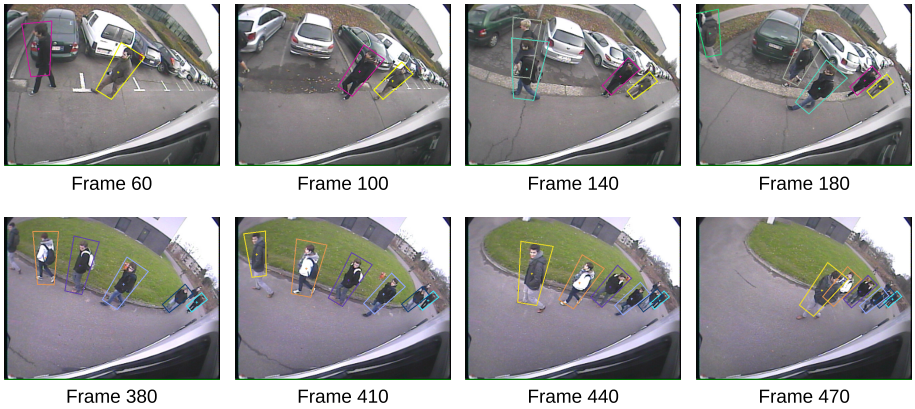


Figure 7.10: Qualitative tracking sequences over two of our datasets (top and bottom row) - see `http://youtu.be/gbnysSoSR1Q` for a video.

$x_k = \begin{bmatrix} x & y & v_x & v_y & s \end{bmatrix}^T$. We employ this scale information as follows. As mentioned we rewarp each pedestrian to a fixed height of 160 pixels, since experiments indicated that this height achieves the optimal accuracy. However, for specific pedestrians this might not be the case. After detection we evaluate the height of the corresponding bounding box. If the height of this bounding box deviates from the expected (calibrated) height at that position, we adjust the scale factor in the Kalman filter that is used in the next frames to perform the warp operation. The transition matrix A then becomes:

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.4}$$

Hence, we now employ a constant velocity motion model with a constant scale. For subsequent frames, the height to which an image patch is warped then slightly differs from the optimal height which was determined in subsection 7.1.3. The remainder of the tracking framework remains identical. Figure 7.10 qualitatively illustrates tracking sequences on two of our datasets.

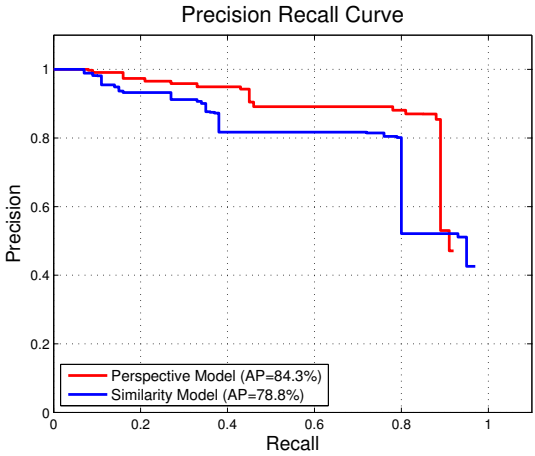


Figure 7.11: Precision-recall curve over our dataset.

7.1.5 Experiments and results

To validate our detection framework we performed extensive experiments concerning both speed and accuracy. For this, we extended our own dataset recorded with a genuine blind spot camera and real truck as presented in chapter 4, subsection 4.2.3. The implementation details and test setup remain identical. For the accuracy and speed experiments presented here, our test set consists of around 4400 labelled pedestrians.

Accuracy

Figure 7.11 displays the precision-recall curve of our algorithm as calculated over our datasets. The red PR curve indicates our novel perspective transformation approach, while the blue PR curve represents our previous similarity transformation approach (evaluated on the same large dataset). As in chapter 4 we assign TPs, FPs and FNs by comparing if an annotation is found in a circular region around the centre of the detection. We notice that, although both achieve very good accuracy results, our novel perspective warping window approach has a clear accuracy advantage over our similarity warping window approach. Indeed, the average precision (AP) for the similarity model equals 78.8%, whereas for the perspective model $AP = 84.3\%$. The accuracy of the similarity model given here is lower as presented in chapter 4. This is due to the fact that we extended our dataset, and thus more test data was evaluated.

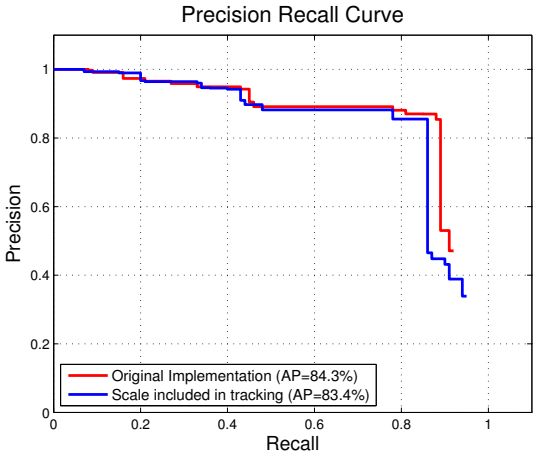


Figure 7.12: The accuracy when including the scale in the Kalman tracker as opposed to the original implementation.

With the perspective model, at a recall rate of 90%, we still achieve a precision of about 85%. Such high accuracy results are due to our warping window approach. Since we know the scale at each position, the number of false positives is minimised. Furthermore this allows to use a sensitive pedestrian detection threshold. Figure 7.12 illustrates the accuracy when the scale is included in the state vector of our Kalman tracker, as compared to the original implementation. As seen, including the scale in our tracking framework has no positive impact on the accuracy – it even slightly lowers the detection accuracy. This is likely due to the fact that the deformable part model inherently is invariant to slight height variations.

Speed

As mentioned in subsection 7.1.3, if used out-of-the-box the baseline pedestrian detector takes 640 ms per frame. Since in our framework we only need to perform detection at a single scale and ROI, the calculation time drastically decreases. For each default search region and tracked pedestrian in the image we need to perform a warp operation and detection. Thus, the total calculation time evidently depends on the number of tracked pedestrians per image. Figure 7.13 displays the average calculation time per ROI. Note that if a detection is found, the average calculation time equals 18.3 ms, while if no detection is found the average calculation time drops to 10.8 ms. The calculation time per region is independent of the position in the image. This was not the case for our original

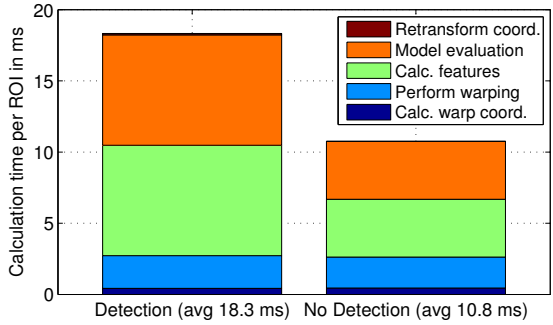


Figure 7.13: Calculation time per ROI.

implementation presented in chapter 4 where the calculation time depends on the rotation angle. There, we first extract each image patch and then perform a rotation operation resulting in an increase of the image area. For this implementation we optimised the extraction of each image patch to avoid this issue.

The average detection time per ROI is subdivided into five steps: the calculation of the warp coordinates, the time needed to perform the warp operation, calculation of the HOG features, evaluation of the pedestrian model and finally the retransformation of the detected coordinates to the input image. The total warp time (*calc. warp coord.* and *perform warping*) only equals about 3 ms. Most time is spent on the actual pedestrian detection. The time needed to perform the retransformation of the coordinates is negligible. Figure 7.14 displays the frames per second as a function of the number of tracked pedestrians we reached on our datasets. If no pedestrians are tracked we achieve 28.2 FPS. On average we achieve 13.0 FPS (with an average of 3.4 pedestrians), while our worst-case frame rate equals 7.0 FPS.

7.1.6 Conclusion

In this section we presented a first extension of our warping window approach. We evaluated two methodologies to increase the detection accuracy. First we modelled the warping operation as a perspective transformation. To construct this perspective transformation we employ the vantage lines in the undistorted camera image. Our experiments indicate that this perspective transformation significantly increases the detection accuracy. Second, we performed experiments to evaluate if a rigid detector could be used in our framework and evaluated if

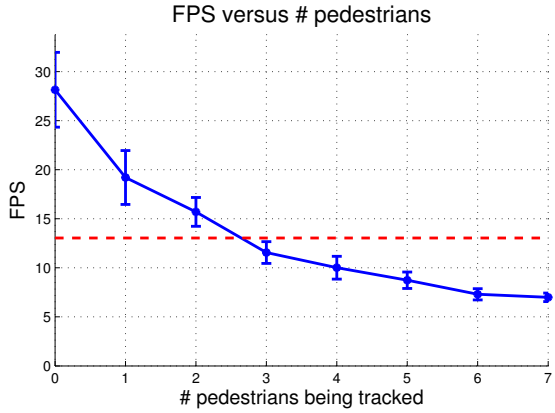


Figure 7.14: Speed performance versus the number of tracked pedestrians (dashed red line indicates the average FPS).

a combination of multiple pedestrian detectors further increases the detection accuracy.

In the next section we present a second extension of our framework towards multiclass detection, thus enabling the simultaneous detection of pedestrians and bicyclists.

7.2 An extension to multiclass detection

7.2.1 Introduction

As mentioned in chapter 2, apart from pedestrians also bicyclists are often the victim of these blind spot accidents. An evident extension of our framework thus is the inclusion of a bicyclist detection methodology.

As a preliminary experiment we qualitatively validate the performance of our initial (pedestrian detection) framework from chapter 4 on a small, preliminary bicyclist dataset. To conduct these experiments no algorithmic changes were performed. Our motivation for conducting these experiments is based on the fact that pedestrians and bicyclists share similar appearance features (e.g. the upper body). Figure 7.15 shows the qualitative experimental results. As one can see in the first few frames (frame 255 and 263), the appearance of the bicyclist is similar to that of a pedestrian. Therefore our algorithm performs well in these situations where the bicyclist is relatively far away. However, in

Figure 7.15: Example output of our tracking algorithm on a bicyclist dataset.

consecutive frames the similarity decreases and the detection is lost. Evidently, a merely direct application of the pedestrian detector is not feasible anymore.

In this section we discuss a more profound methodology to tackle this problem. We present a multiclass detection methodology which enables the efficient detection of both pedestrians and bicyclists in the challenging blind spot camera images. To achieve this we present the integration of our warping window approach with multiple object detectors which we intelligently combine in a probabilistic manner.

The main contributions in this section are two-fold. We give an approach to efficiently combine multiple object detectors using a probabilistic manner for these non-trivial images, and we propose a methodology which selects the most appropriate model to evaluate based on the position in the image.

In a nutshell, our framework works as follows. First we employ our perspective warping window approach as discussed above: at each position in the input image we locally model the transformation due to the viewpoint distortion. Using this information we can rewarp each region of interest, effectively undoing the local distortion. Next we extract image features on this rewarped patch. However, on each rewarped image patch we now evaluate three different detection models. Apart from the standard pedestrian model, we evaluate an upper body model and one of three components (i.e. different viewpoints) of a bicycle model. The specific component is selected depending on the position in the image. This upper body model, combined with a bicycle model enables the efficient detection of bicyclists. For each of these three models we generate a probability map. These probability maps are then combined into a single detection probability map for that image patch. Finally, to cope with missing detections we integrate these detection maps in our tracking-by-detection methodology.

Several works exist which perform bicycle detection specifically for traffic safety applications, and are thus related to our work. However, to the best of our knowledge, often only forward-looking cameras [18, 62] or only stationary cameras [103] are used. The work presented in [104] is closely related to our work.

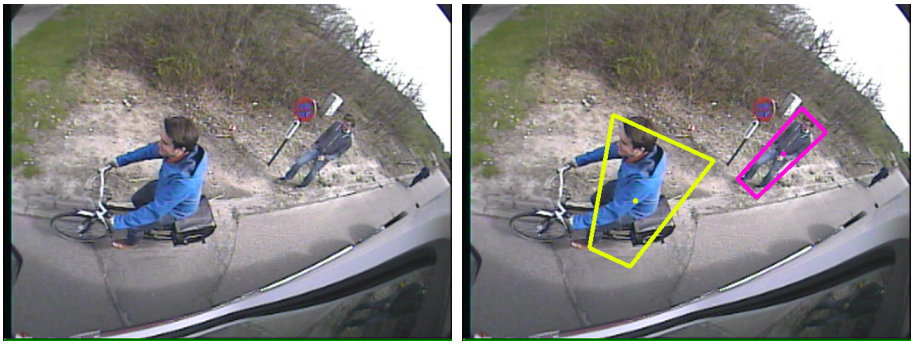


Figure 7.16: Left: Example frame from our dataset recorded with both pedestrians and bicyclists. Right: Output detections of our framework.

Here, the authors also exploit geometrical constraints to perform ROI extraction. They tackle the multi-view problem of bicyclists by dividing them into eight subcategories based on their viewpoint (as proposed in [77]). However, their approach is only applied on forward-looking camera images and only bicyclists are detected. We differ significantly from all of these works: we aim to develop an efficient framework that enables the detection of multiclass objects in camera images with non-standard viewpoint and high lens distortion.

The remainder of this section is structured as follows. In subsection 7.2.2 we provide details on our algorithmic implementation. We discuss our detection framework, indicate which detection models are employed, and how they were trained. We then discuss how we combine all detection outputs into a final classification result. In subsection 7.2.4 we discuss experimental results to validate the effectiveness of our approach. For this, we extended our previous datasets to include both pedestrians and bicyclists. Figure 7.16 displays an example frame of our new dataset (left) and the output of our framework (right). Apart from standard accuracy and speed experiments, we performed experiments to validate the contribution of each of the three detection models. Our algorithm achieves excellent accuracy results on these challenging datasets.

7.2.2 Algorithmic approach

As a start point we employ the same perspective framework as proposed in the previous section. If we ought to run a standard object (e.g. pedestrian) detector on these images they need to be evaluated at multiple locations, scales and orientations which is impossible to compute in real-time. Moreover, due to the viewpoint and high lens distortion the detection accuracy will be suboptimal. We thus exploit the fact that the exact transformation (that is, rotation, scale

and perspective effects) only depends on the position in the image. At each position in the image we locally model this transformation, and rewarp the region of interest to an undistorted, upright and fixed scale image patch avoiding the need to compute a full scale-space pyramid.

However, since we aim to detect both pedestrians and bicyclists, the detection pipeline significantly differs from here on. In the next step we extract features and run multiple detection models on these image patches. To reduce the computational complexity we employ feature sharing and only run specific models at specific locations. These detection maps are then combined into a single probability map. Finally we integrate this information in a tracking-by-detection framework. Figure 7.17 gives an overview of our detection approach. Note that due to space constraints the processing steps are shown for two ROIs only, in practice much more ROIs are validated. We now discuss each of the consecutive steps of our detection pipeline in detail.

Warping patches

In a first step we employ our perspective warping window approach. For details, we refer to subsection 7.1.2. We modelled the local distortion as a perspective transformation. The local deformation for each position is extracted in an offline step, and stored in two *deformation maps* as visualised in the left of figure 7.17. We thus model the pedestrians as planar objects, faced towards the camera. Our experiments indicate that this is a valid assumption for pedestrians. For bicyclists, this assumption is not valid at all positions in the image. However, this concern is tackled further in our detection pipeline: we evaluate multiple bicycle viewpoint models depending on the position in the image.

Evidently, the viewpoint of a bicyclist also depends on its relative orientation with respect to the truck. Keep in mind that we aim to develop a safety system for blind spot accidents. These all occur in a similar manner: a bicyclist (or pedestrian) continues its way straight ahead while the truck driver takes a right-hand turn. As such, in the remainder of this section our main goal is the detection of bicyclists which are oriented parallel to the truck. During detection we employ the aforementioned deformation maps and the vantage point to effectively undo the local rotation and perspective transformation, and warp the ROIs to a fixed scale of 140 pixels, as this has proven to be an adequate trade-off between accuracy and computational complexity.

In the previous section we utilised a fixed scale factor of 160 pixels. However, since here we need to evaluate multiple detection models the computational complexity is significantly higher. Furthermore, our experiments indicated that – when lowering the detection scale from 160 pixels to 140 pixels – the accuracy

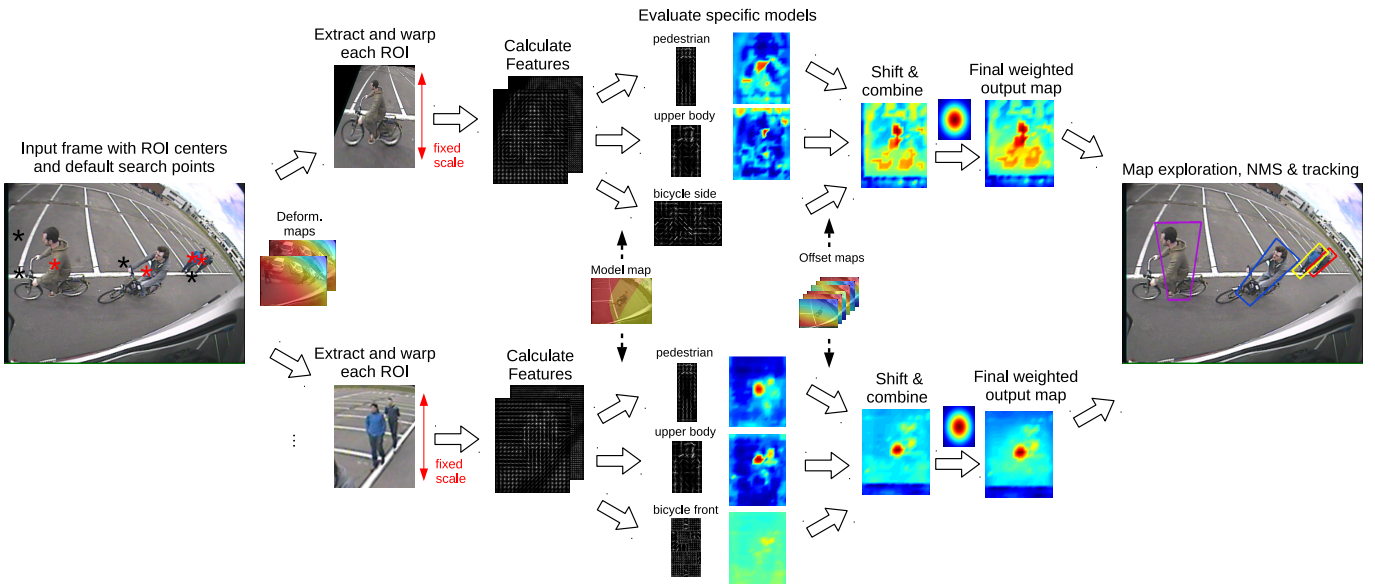


Figure 7.17: Overview of our algorithmic approach. Note that the probability maps displayed here are interpolated for visual purposes; in practice they are calculated discretely on a grid with a step size of 8 pixels.

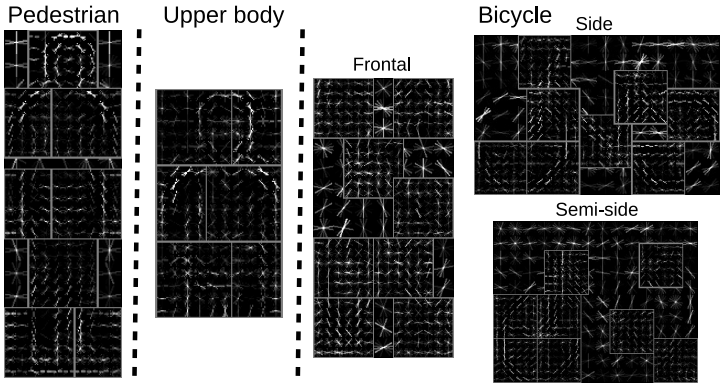


Figure 7.18: The different detection models with their respective components. The pedestrian and upper body model consist of a single component (viewpoint). The bicycle model consists of three components: a frontal, semi-side and side component.

drop is minimal (see figure 7.5). These upright, fixed scale image patches are then fed in to our detection pipeline.

Object detection pipeline

The unwarped image patches can now be processed to detect both pedestrians and bicyclists. Currently, rigid detectors are slightly more accurate as compared to deformable part-based model approaches [7]. However, the advantage of the latter is of strong importance in our framework: since deformation is allowed, slight deviations from the trained model and the object to be detected are tolerated. This invariance to slight variations in height was proven in subsection 7.1.3. Since we only perform detection at a single scale this deformation is essential: multiple scales are needed with a rigid model to achieve accurate detection results. Therefore we opted to use the cascaded DPM [43] as a baseline. In a first step, for each ROI image patch we extract a 31 dimensional feature vector (consisting of HOG and contrast features). Since the detection accuracy increases if the features for the different parts are calculated more densely [44], this is done for two different bin sizes. To robustly detect both pedestrians and bicyclists we share these features between different detection models.

At each position in the image we validate three models: a pedestrian model (trained on INRIA), an upper body model and a bicycle model (both trained on the VOC 2009 dataset). The pedestrian and upper body model consist of a single component (i.e. a single viewpoint). However, the bicycle model

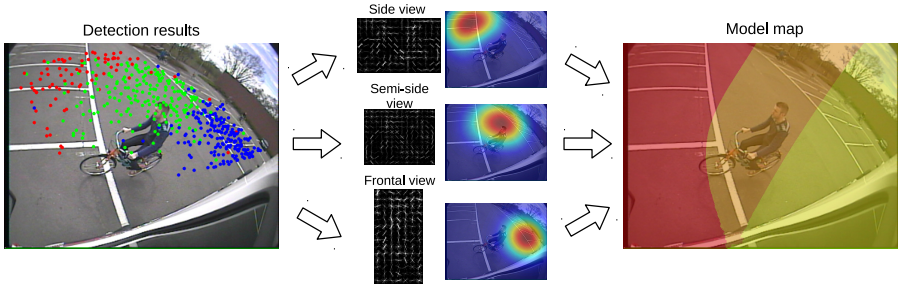


Figure 7.19: Generation of the *Model map* which indicates which bicycle component should be evaluated where in the image.

consists of three components (corresponding to three different viewpoints): a frontal, semi-side and sideways looking viewpoint. Figure 7.18 displays our three detection models. Where needed we retrained the models such that the size of their root models are equal. This ensures that all models have a maximal response at the same image resolution, and is needed since we perform detection on a single scale. Details on this are discussed further in this section.

Based on the position in the image we perform *model selection*: we only select the single, most optimal bicycle detection component to run at that location and thus decrease the calculation time. For this, in an offline phase we evaluated all three components on labelled bicyclists homogeneously spread over the image and selected the best scoring component for each image position. This is illustrated in figure 7.19. The left figure displays where each component performed best (red: side view component, green: semi-side component, blue: frontal component). With these data we generated a probability map for each component and combined these maps into a final image segmentation, as shown in figure 7.19. This final *Model map* thus indicates which bicycle component should be evaluated at each position. These three models (pedestrian, upper body and one of the bicycle components) - and their mirrored versions (we take the maximum of both) - are evaluated on a grid of 8 pixels, and yield three discrete probability maps for each image patch.

Evaluating the different detection models

In this subsection we briefly discuss the different detection models that we employed. For the pedestrian detection model we used the pre-trained deformable part model included with the publicly available vanilla DPM implementation [45]. This detection model was trained on the INRIA

dataset [21], consists of 8 parts and has a size of 120×40 pixels. As for the upper body model several publicly available implementations exist. As an example, [38] propose experiments with both a HOG-based and DPM-based upper body model for the task of human pose estimation. In our framework we employ the deformable part upper body model as presented in [22]. Here, the model was trained on the VOC 2008 dataset [42]. For training, about 4200 positive images (and their mirrored version), and about 1000 negative images were used. This detection model consists of 8 parts and three different components ranging from a full upper body component to a detection component which only consists of the head and shoulders of a person.

In our application we only employ this full upper body component as detection model (as displayed in figure 7.18). The detection model has a size of 54×40 pixels. Thus, the size of this detection model is consistent with the pedestrian model that we use, and no retraining was needed. This was to be expected, since the authors trained this model using only the upper 60% of the labelled bounding boxes of the full human bodies. To get an indication of the detection accuracy of such an upper body model we performed experiments to validate the accuracy of this upper body model with respect to a full person (pedestrian) model. Evidently, one would expect lower detection performance (since less information is used to perform detection). To verify this, we compared the accuracy of several pedestrian detection models and the upper body model that we use in our detection framework on the INRIA test set. Note that this test set evidently differs from the INRIA training set used to train e.g. the pedestrian DPM model. The INRIA test set consists of about 300 images containing mostly large scale (high resolution) pedestrians – occasionally a bicyclist is included. As such, this is a relatively *easy* dataset. These results are displayed in figure 7.20.

For this comparison we utilised four pedestrian detectors: *ACF*, *HOG*, *VJ* and the original cascaded DPM model (*Latv4-cc-original*). The remaining curves indicate the accuracy performance of our upper body detector when varying the degree of overlap used to perform non-maxima suppression (ranging from 50% to 0%). As seen, both ACF (AP=89%) and the full body DPM model (AP=88%) achieve excellent accuracy results. Traditionally (as in the original DPM model here), NMS is performed when detections overlap for at least 50%. This achieves optimal accuracy results for the pedestrian detectors. Our experiments indicated that this design parameter is more sensitive for the upper body detector.

The upper body detector achieves an AP of 76% when using the traditional overlap parameter. Although the accuracy is lower as opposed to the pedestrian detectors, despite the smaller detection model the accuracy remains good. When removing detections with smaller overlap, the detection accuracy increases. An optimal value of the NMS parameters needs to be determined. If no detections

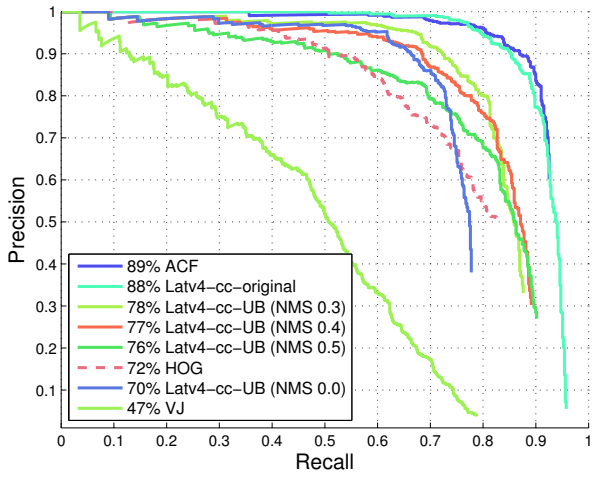


Figure 7.20: Comparing the accuracy of the upper body (UB) detection model with full pedestrian detection models.

are removed, all pedestrians are detected multiple times resulting in a high amount of *false positives*. If detections are removed with only slight overlap (e.g. 0% in the most extreme case) pedestrians standing close to each other are never detected and the accuracy will decrease significantly. The optimal value for our upper body detector is found at around 30%. In this case, the detection accuracy increases to 78%. We assume that this is due to larger localisation errors. Since the upper body detection model is much smaller, the resulting detected bounding boxes deviate more from the exact labelled position.

A publicly available deformable part model for bicycles exists [45]. However, during training the model size is automatically determined based on the available training images. The size of this detection model is inconsistent with those of our pedestrian and upper body model. Therefore, we trained a deformable part bicycle model using the VOC 2009 training set [42] with model sizes compatible with our one-scale only framework. We utilised 468 positive images (and their mirrored version), and 3300 negative images. The model again consists of 8 parts. As mentioned, three different viewpoints were trained. For this, the annotated image patches used for training are separated based on their aspect ratio. The size of e.g. the side component equals 72×112 pixels. To validate the effectiveness our bicycle detection model, we compared the accuracy with the publicly available model on the VOC 2009 test set. These results are visualised in figure 7.21. The accuracy of our retrained bicycle model is higher than the original DPM bicycle model. This is due to the large model resolution: the

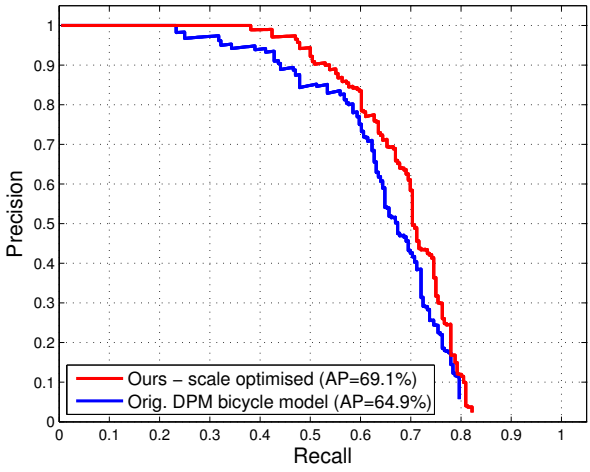


Figure 7.21: Comparing the accuracy of our scale-optimised bicycle detection model (red curve) with the original bicycle model (blue curve).

publicly available model has a slightly lower resolution.

Combining probability maps

In a final step these probability maps $P_i(\mathbf{x})$ with $\mathbf{x} = [x, y]$ for each component i are combined into a single probability map using:

$$P_{final}(\mathbf{x}) = \max_{\substack{\mathbf{x} \\ i \in \{1,2,3\}}} (P_i(\mathbf{x}) - d_i(\mathbf{x})) + G(\mathbf{x}) \tag{7.5}$$

Here, $d_i(\mathbf{x})$ indicates an offset for each component used to ensure correct detection localisation. We shift each map such that the expected maximum of the detection models coincides with each other. Indeed, as seen in figure 7.22 the maxima of the detection models per default do not coincide. For this, the upper body model is shifted downwards, and the bicycle model is shifted upwards. This exact offset again depends on the position in the image, and is extracted simultaneously with the generation of the model map. They are visualised in figure 7.17 as the *Offset maps*. To emphasize the centre location the map is weighted with $G(\mathbf{x})$, centred at the image patch:

$$G(\mathbf{x}) = \alpha \left[e^{\left(-\frac{\mathbf{x}^2}{2\sigma^2}\right)} - 1 \right] \tag{7.6}$$

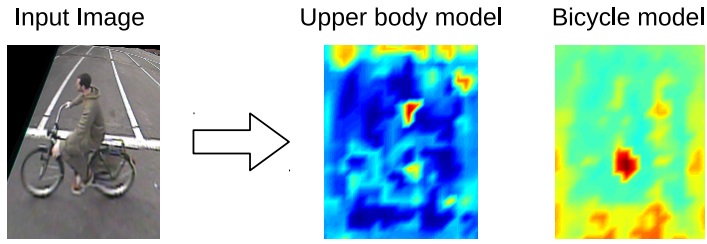


Figure 7.22: The probability maps of the upper body model and one component of the bicycle model for a specific image patch. As seen, both maxima are shifted.

Where α indicates the penalty at the image borders (we empirically determined $\alpha = 2$). Taking the maxima of all detection models is allowed. During training an additional bias is added to ensure that the returned detection scores are comparable.

7.2.3 Map exploration, NMS and tracking

Finally, these ROI probability maps are integrated in our tracking-by-detection framework to improve the accuracy and detection speed. This is done as follows. We define default search points (corresponding to search ROIs) at strategic positions in the image with respect to the blind spot zone. The exact positions of these points are discussed in the next section. These positions are evaluated every frame using our pipeline mentioned above. Each probability map is then thresholded to extract local maxima.

Next we perform *non-maxima suppression* (NMS) to cope with overlapping detections - using a variant of the 50% intersection criterion [35] - keeping only the best scoring detections. For each new detection a Kalman filter is instantiated. For consecutive frames, we predict the future location of the tracked instances, and use these predicted ROI centres (together with the default search points) as input to our detection pipeline. The implementation details of this tracking framework are identical to subsection 7.1.4. A qualitative tracking result is shown in figure 7.23.

7.2.4 Experiments and results

We performed extensive experiments to validate both the accuracy and speed of our algorithm. In previous experiments our dataset only consisted of pedestrians.

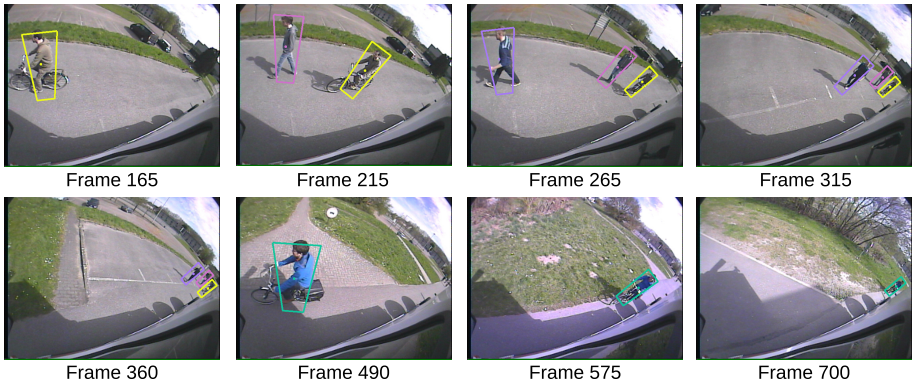


Figure 7.23: A qualitative tracking sequence over one of our datasets. See `http://youtu.be/0xFdDOYxKK8` for a video.

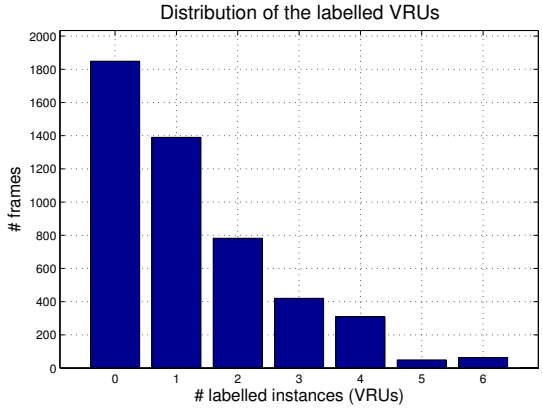


Figure 7.24: The distribution of the labelled VRUs over our datasets.

Here, to evaluate our multiclass detection scheme we recorded several new simulated dangerous blind spot scenarios with a genuine blind spot camera mounted on a real truck involving both pedestrians and bicyclists. In total seven different scenarios were recorded each in which the truck driver makes a right turn, and the vulnerable road users act differently (e.g. the truck driver notices the VRUs and lets them pass, or the truck driver keeps driving simulating a near-accident). Our total test set consists of about 5000 frames, in which over 3600 pedestrians and 2400 bicyclists were manually labelled. Figure 7.24 displays the distribution of the VRUs over our dataset.

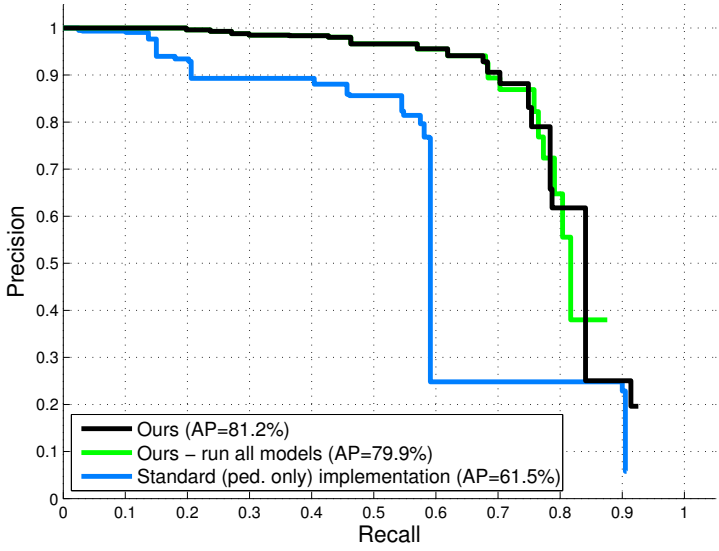


Figure 7.25: The accuracy results of our algorithm (black curve) compared with an implementation where all detections models are evaluated (green curve) and our previous – pedestrian detection only – framework (blue curve).

Our implementation methodology and test hardware remains the same as to previous chapters: our framework is mainly implemented in Matlab with time-consuming parts (e.g. the detection and homography) in both C and OpenCV, and the hardware consists of an Intel Xeon E5 CPU at 3.1 GHz. As default search regions we define two entry points at the left, one entry point at the end of the truck, and one point in the blind spot zone (to recover lost tracks). These default search regions are indicated with a black asterisk (*) in the leftmost in figure 7.17.

Figure 7.25 displays the accuracy of our algorithm using a *precision-recall* curve (black curve). We achieve excellent accuracy (Average Precision of 81.2%). We also compare our algorithm with the standard (i.e. pedestrian detection only) implementation of our perspective warping window framework presented in section 7.1 (blue curve). As seen, our algorithm easily outperforms this framework on these datasets that also contain bicyclists. The green curve plots the accuracy when the model selection is discarded, and thus all detection models are evaluated at each location. Evidently, running all models increases the computation time. The accuracy difference is minimal, indicating that our model selection procedure using the *Model Map* of figure 7.19 is optimal. We thus achieve the same detection accuracy at lower computational complexity.

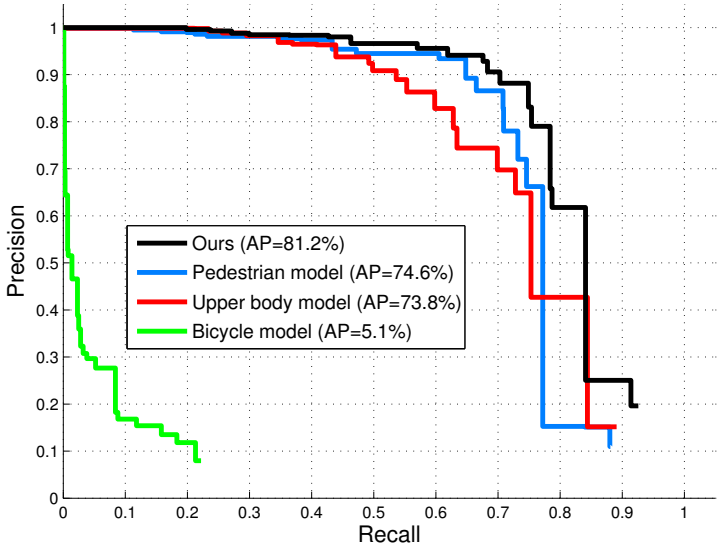


Figure 7.26: The accuracy of our approach (black curve) compared to the accuracy when only individual detection models are employed.

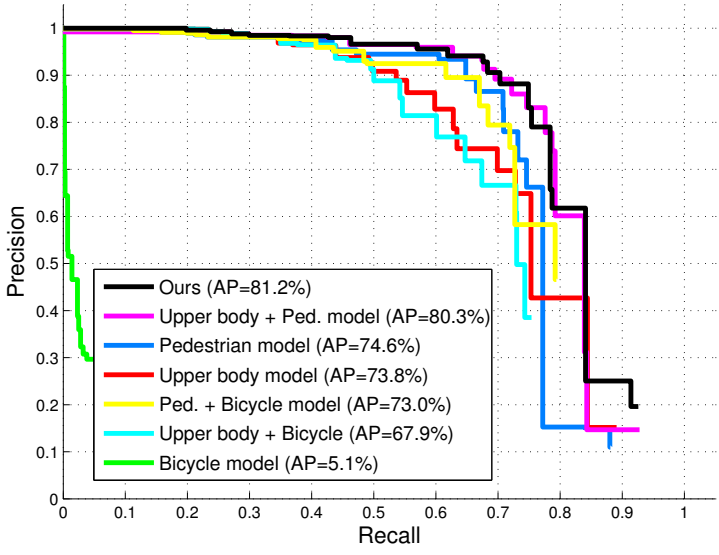


Figure 7.27: Accuracy of all possible combinations with the three detection models.

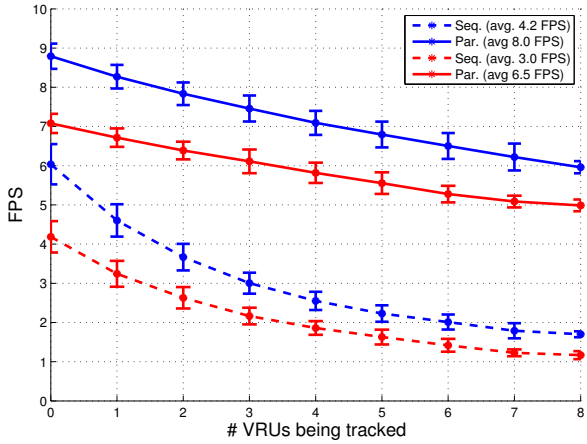


Figure 7.28: Processing speeds of our algorithm.

Next we compared the accuracy when running only individual models with the accuracy that our VRU detection framework achieves, shown in figure 7.26. Evidently, when only evaluating the bicycle model the detection accuracy is low. As seen, the accuracy significantly increases when all three detection models are combined.

To validate which model accounts for the highest gain in accuracy we performed experiments in which all possible combinations of these three detection models were evaluated. These results are displayed in figure 7.27. An interesting observation is noticed. The combination of all three models achieves the highest accuracy. However, the combination of only the upper body model and the pedestrian model achieves only a slightly lower accuracy. Two possible explanations for this phenomenon exist. A first assumption is that bicyclists are equally well detected with only the upper body model. The use of the bicycle model might be unnecessary. However, such claims need further investigation. It is also possible that too few difficult to detect instances (i.e. VRUs at positions where large distortion occurs) are available in our dataset. To validate this, we ought to increase the size of our dataset and include such specific scenarios.

We performed several computational speed experiments. Although we mainly focused on high accuracy, during algorithmic development we aimed at keeping computational complexity (within the limitations of Matlab) as minimal as possible. Figure 7.28 displays the execution speed in function of the number of tracked VRUs in a frame. Evidently, the computation time increases with multiple tracks. To partially cope with this, we implemented both a sequential (dashed lines) and a parallel version (solid lines) of our framework. Parallel

processing is achieved by processing each search region in a separate thread. The blue curves indicate our implementation with model selection, the red curves indicate processing speeds when evaluating all models.

Processing only specific models increases the speed with about 25%, with no loss in accuracy. Our best implementation achieves an average processing speed of 8.0 FPS on our realistic dataset. The processing speed thus is lower as opposed to our original perspective implementation. This is evident since multiple detection models are evaluated. Furthermore, to generate the probability maps a score is extracted at each location. For this, no early pruning in the cascaded deformable part implementation is possible. Please note, however, this proof of concept implementation was developed mainly in Matlab. Specific time consuming parts were implemented in C. However, the interfacing between both software instances causes a significant delay (due to e.g. different data formats). Thus, a significant gain in processing speed remains achievable.

7.2.5 Conclusion

In this section we presented a second extension of our warping window approach: we extended our framework to cope with the detection of bicyclists. We integrated our warping window approach with an efficient multiclass object detection scheme. For this, we developed a probabilistic detection framework in which we combine the detection results of three different detection models: a pedestrian model, an upper body model and a bicycle model. Note that in the current implementation these detection results are combined in a naive manner. Indeed, a more profound combination methodology (e.g. the combination methodology as presented in chapter 6) might further increase the detection accuracy. We were unable to integrate this combination approach in our probabilistic framework due to time constraints. Since detection is performed at a single scale only, we retrained specific models to obtain equal model sizes. Accuracy experiments were performed to evaluate the performance of these retrained models.

To reduce the computational complexity we only run specific viewpoint detectors based on the position in the image. We performed extensive accuracy experiments and evaluated the accuracy of all three individual models and each possible combination. Our framework achieves excellent accuracy, while keeping the computational complexity adequate for practical applications. As mentioned, several speed optimisations are applicable to increase the detection speed.

7.3 Conclusion

In this chapter we presented two extensions of our initial pedestrian detection and tracking framework (presented in chapter 4). Both extensions are presented in a distinctive section in this chapter.

In the first part of this chapter we present an extension which increases the detection accuracy. For this, we revised the *warping* stage of our detection framework: the transformation was modelled as a perspective distortion. Furthermore we analysed the performance of a rigid pedestrian detector in our framework, and evaluated if a combination of both pedestrian detectors could further increase the detection accuracy.

In the second part of this chapter we extended our framework to a multiclass detection framework. Apart from pedestrians, this approach allows to efficiently detect bicyclists as well. For this, we integrated our warping approach in a probabilistic object detection scheme. We run multiple detection models, and combine them into a single final probability map. We performed extensive experiments to validate the performance of each individual detection model, and all possible combinations.

In the next chapter we finalise our detection framework. We tackle the non-deterministic calculation time, evaluate the use of multiple detection scales (allowing to also detect children), and perform system level tests.

Chapter 8

A final complete safety system

The final goal of this dissertation is the development of a complete active blind spot alarm system, solely relying on the blind spot camera as input. In the previous chapter we presented a VRU detection and tracking framework which achieves excellent results. In this chapter we polish this framework and elevate it towards such a complete final active alarm system.

For this, we first tackle two limitations of our approach. Currently, the calculation time of our framework depends on the number of VRUs which are being tracked. Here, we propose an approach that ensures a deterministic calculation time. This evidently is of crucial importance for these active safety systems. Next we present an approach that extends the applicability of the framework from only adult pedestrians and bicyclists to children.

Finally, we elevate our framework to an active alarm system and perform accuracy results at *system level*. In chapter 2 we discussed the specifications that such a system should achieve. Here, we now validate each of these required specifications with respect to our final system, and discuss the usability of our system when employed in real-world scenarios.

Part of this chapter was published at the ECCV CVRSUAD workshop in 2016 [110].

8.1 Introduction

Throughout the previous chapters we developed an efficient VRU detection and tracking framework. In chapter 4 we initially started from a basic tracking-by-detection framework developed for backward-looking images recorded from a standard moving car. We then presented our warping window approach to cope with the viewpoint distortion induced when using a genuine blind spot camera. This framework was further refined in chapter 7 where we proposed two extensions. A first extension focused on an increase in detection accuracy. For this, we exploited the perspective transformation induced by our specific viewpoint and experimented with multiple pedestrian detectors. The second extension enabled the detection of bicyclists. For this, we developed a probabilistic multiclass detection and tracking methodology.

Our final VRU detection and tracking framework presented in chapter 7 is able to detect and track both pedestrians and bicyclists with high accuracy at reasonable processing speeds. However, in its current form this framework has two caveats if used as a blind sport alarm system.

Firstly, the calculation time is non-deterministic. Our tracking-by-detection framework relies on initial search coordinates which are defined at strategic positions in the image. When VRUs are detected at these positions, a new track is instantiated and evaluated in the consecutive frames. This approach thus implies that the processing speed depends on the number of tracks that are evaluated. Such non-deterministic behaviour is not suited for hard real-time applications where *latency* and *processing speed* are of crucial importance. We must be able to guarantee that the system reacts within a constant time.

Secondly, in its current form the framework is only developed with adult pedestrians and bicyclists in mind. We perform detection at only a single scale, which is trained in an offline phase. As proven in previous chapters, the deformable part models are relatively invariant to small variations in height. Pedestrians and bicyclists which slightly diverge from the calibrated height are still easily detectable. However, if a large scale variation exists between our calibrated height and the *objects* that need to be detected our system fails. This is the case for children, which evidently do not comply with our trained scales.

In this chapter we propose two solutions for both aforementioned problems: in section 8.2 we present a methodology which tackles the non-deterministic behaviour of our framework. Section 8.3 then discusses our proposed solution towards the detection of children.

The next two sections of this chapter focus on our framework at *system level*. In previous chapters we developed a detection and tracking framework for VRUs in

blind spot camera images. In section 8.4 we elevate our tracking approach and convert it into a final active alarm system. We present accuracy experiments when viewed as an active alarm system. Evidently, the usability and acceptance of such an active safety system in real-life scenarios is influenced by many factors such as the false alarm rate, the latency (i.e. reaction time), the detection speed and ease of use of such a system. In chapter 2, section 2.5 we discussed the required specifications that such a system should achieve to be usable in practice. Here, in section 8.5 we validate if our final developed active safety system meets all of these stringent demands. Finally, we conclude this chapter in section 8.6.

8.2 Deterministic calculation time

8.2.1 Introduction

Our tracking-by-detection framework inherently implies a non-deterministic calculation time. At strategic locations in the image we define initial search coordinates and start a new track for each VRU that is being detected in these initial regions. The exact calculation time currently thus highly depends on the number of started tracks, i.e. the number of tracked pedestrians in the image. Therefore, in previous chapters we consistently reported the calculation time in function of the number of tracked pedestrians or VRUs.

Such non-deterministic behaviour is unacceptable for hard real-time safety applications as developed here, where an upper bound with respect to the calculation time must be guaranteed. Indeed, these safety systems are only usable in practice if the latency is predictable. Furthermore, a deterministic calculation time allows for a more efficient implementation. For example, an optimal throughput is easier obtained if the exact calculation time of each stage in the detection pipeline is fixed and known. This is especially the case for multi-threaded or hybrid applications where, if each step in the processing pipeline is matched towards equal calculation time, the throughput is maximised.

We exploited this knowledge in our hybrid CPU/GPU implementation which achieved a detection rate of 500 Hz (see chapter 4, section 4.3). Such implementations maximally exploit the hardware capabilities of the platform on which they are executed. This allows for real-time operation, even on low-cost embedded platforms. We illustrated this with our real-time demonstrator which achieves pedestrian detection in these blind spot camera images at 15 frames per second on such a platform (see subsection 4.3.5).

In this section we thus propose a methodology that copes with this non-deterministic behaviour. The remainder of this section is as follows. In subsection 8.2.2 we discuss our approach that achieves deterministic calculation times. We present experiments with respect to the influence on the accuracy and computational complexity of this approach in subsection 8.2.3. Finally, we conclude this approach in subsection 8.2.4.

8.2.2 Approach

In our final detection framework we tackle this non-deterministic behaviour as follows. We first define a blind spot zone in the image in which all pedestrians and bicyclists ought to be detected. Determining this zone correctly is of crucial importance for the effectiveness of our final alarm system. A strong correlation exists between the size of this detection zone and the latency of our final detection system. As mentioned in chapter 2, most accidents occur when the truck makes a right turn without noticing pedestrians or bicyclists that continue their way straight ahead. Research indicates that it takes a worst-case reaction time of about 1.5 seconds for a truck driver to react when confronted with an event and to undertake the effective break action [19]. Thus, an early detection of the VRUs is crucial.

For this, a large detection zone is needed, which ideally starts far behind the truck itself. In such scenarios e.g. fast moving bicyclists are detected early and enough time remains for the truck driver to interpret the alarm signal and undertake a corresponding action. However, such a large detection zone requires significant calculation power. Indeed, the size of this blind spot detection zone essentially determines a trade-off between the latency of our alarm system and the required computation power. Furthermore, the allowed latency depends on the relative speed of the bicyclist and the truck itself. To perform our consecutive validation experiments, we constructed a detection zone as illustrated in figure 8.1.

The red zone defines our blind spot detection zone (on the ground plane). All VRUs which enter this zone ought to be detected as soon as possible. The size of the zone is approximately 6.60 metre by 2.60 metre. Since our implementations presented in previous chapters perform the transformation based on the centroids of the pedestrians and bicyclists, we defined the slightly larger and higher positioned detection zone displayed in green in figure 8.1. If we ensure detection of pedestrians and bicyclists when their centroid enter this green zone, this approximately translates as entering the red zone with their contact point at the ground plane. Further experimental results discussed in this chapter are based on this delineated zone. In this section we now only

Figure 8.1: Our defined blind spot detection zone. The red zone indicates the zone in which VRUs need to be detected. The green zone indicates the region in which we position our search coordinates.

discuss accuracy and speed results. The consequences of this defined zone with respect to the reaction time and latency of the alarm system are discussed further in section 8.5.

To perform VRU detection with deterministic calculation times we need to avoid the dependency on the number of VRUs in the image because of these initial search regions and the creation of new search coordinates for each tracked pedestrian or bicyclist. Therefore, we now choose fixed search points within this previously defined (green) blind spot zone. At each of these search points we perform the exact same approach as discussed in chapter 7, section 7.2. For each search point we employ our perspective warping window approach, warp this region to upright and undistorted image patches and then evaluate all three detection models (pedestrian model, upper body model and one of the three bicycle components). The remainder of the detection pipeline is identical to figure 7.17 on page 136. Note that we still employ the tracking-by-detecting approach to cope with missing detections and to increase the robustness. However, we do not utilise the predicted Kalman future locations as input to our warping framework. These future locations are now only used to match detections from the previous frames in future frames. This remains needed to cope with missing detections (e.g. in between two search points).

Since now the number of search points is fixed, the calculation time becomes deterministic. We positioned these search points on a linear grid distributed in the above mentioned green detection zone. Evidently, the number of grid points

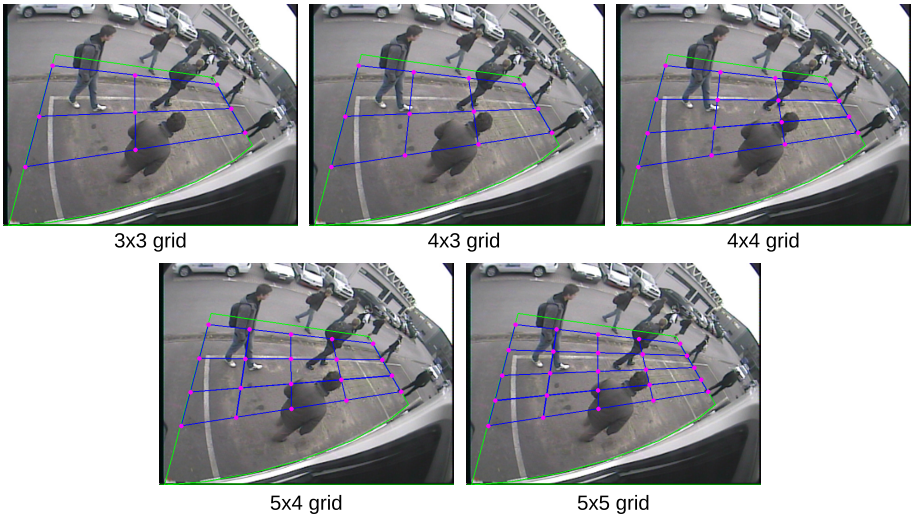
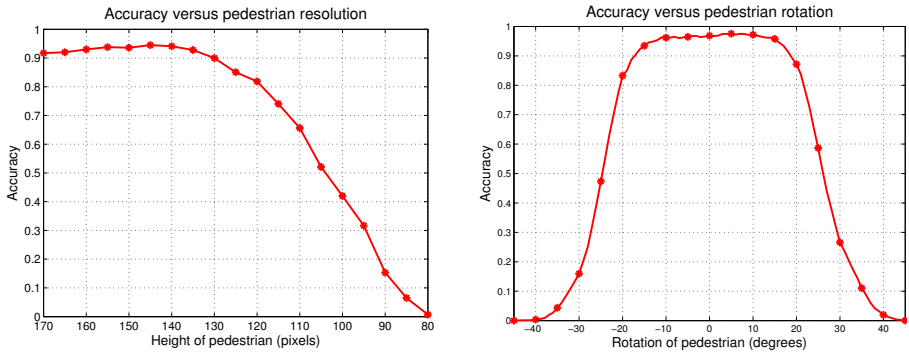


Figure 8.2: An overview of the different detection grids used in our final detection scheme. The magenta points indicate the search points.

determines a trade-off between the computational complexity and the accuracy. To determine this trade-off we evaluated the performance of five different grids, ranging from dense to fine: a 3×3 grid, a 4×3 grid, a 4×4 grid, a 5×4 grid and finally a 5×5 grid. These grids are visualised as the magenta points in figure 8.2. For our real-time demonstrator presented in chapter 4 we utilised a similar strategy with fixed search points.

However, due to the non-linear distortion and specific viewpoint a linear grid might not be the most optimal distribution of these search points. For example, more search points could be used at locations where the perspective distortion is more pronounced. To validate these assumptions we define an additional detection grid in which we automatically distribute the search points based on the degree of distortion in the image. For this, we segment the blind spot zone in areas based on similar rotation and scale values, as seen in figure 8.4. To determine these ranges, we need to evaluate the invariance of our deformable part detector with respect to both rotation and scale variation.

The invariance with respect to the scale variation has been evaluated multiple times – for different datasets and transformation methodologies – in previous chapters in this dissertation. Take for example our experimental results with respect to the height variation performed in chapter 4, subsection 4.2.2. There, we extracted about 1000 pedestrians, rescaled them to fixed heights and



(a) Accuracy versus height (pixels).

(b) Accuracy versus rotation (degrees).

Figure 8.3: The detection accuracy of the deformable part model on pedestrian patches versus both different heights and degrees of rotation.

determined the accuracy. For clarity, these results are visualised in figure 8.3 (a). As seen, slight variations on the exact height of 140 pixels cause negligible loss in accuracy. We performed similar experiments with respect to the variation in rotation. For this, we evaluated the detection accuracy of our deformable part pedestrian detector on rotated versions of about 6000 pedestrian image patches. These accuracy results are visualised in figure 8.3 (b). A similar observation is noticed. Small rotations have almost no effect on the detection accuracy.

This information is combined into final detection points as follows. We first segment the delineated blind spot zone in distinctive areas with similar ranges for both the rotation and scale. Based on the experiments above we allowed for a variation in scale of 20 pixels and rotation of 10 degrees. Extremely small scale values are rejected. These segmented areas are visualised in the left two images of figure 8.4. Next, we combine both segmentations into final segments as visualised in the right image of figure 8.4. Finally, we automatically place grid points based on the size of each remaining region, visualised as the red dots. Extremely small regions were rejected. Small regions were assigned a single search coordinate in their centre of gravity. For the larger regions we distributed multiple search coordinates on the line constructed through their respective centre of gravity and the vantage point. We position two search points on this line. One point is centred in the specific region whereas the second point is placed at about 10% of the region border (furthest away from the vantage point). This results in a grid of 12 points, which we coined the *dynamic grid*.

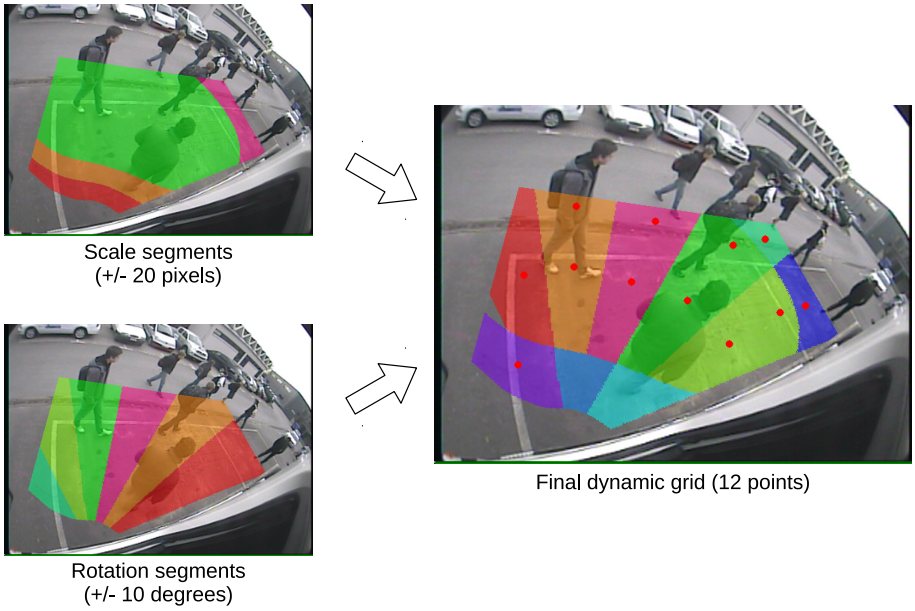


Figure 8.4: Construction of the dynamic grid. For this, we segmented the blind spot zone in areas with similar rotation and scale values.

In the next section we now discuss the influence of each of these six detection grids on both the accuracy and the detection speed.

8.2.3 Experiments and results

Our experiments were performed on an extended version of the dataset as discussed in chapter 7, subsection 7.2.4. We further increased the size of this dataset with additional sequences (and thus labelled instances). Our test set now consists of about 5500 frames, in which about 8000 VRUs were labelled. Since we now only detect VRUs in this delineated blind spot zone, we evidently discard all annotations outside this zone. Figure 8.5 illustrates the influence of this assumption on the distribution of the number of labelled instances in our dataset. About 42% of all annotations are maintained. However, this is a slight underestimate of the real number of labels that are used during test time as we will discuss below.

We performed experiments with respect to both the detection accuracy and detection speed of all six different grids. Additionally, we performed accuracy

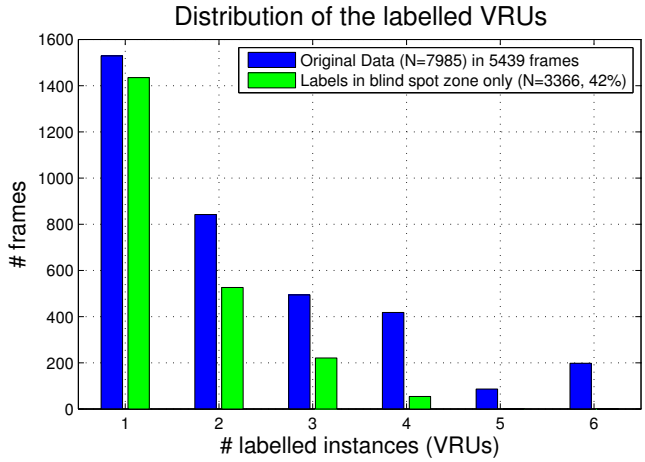


Figure 8.5: A comparison between the original distribution of the labelled VRUs and when only those in the blind spot zone are retained. Blue bars: Original labels for our dataset. Green bars: Labels in blind spot zone only.

experiments to determine the influence on the detection accuracy of our final algorithm from chapter 7 when only performing detection in this predefined blind spot zone.

Accuracy when using grids

To perform our accuracy experiments in a fair manner, we need to tackle the following problem. As discussed above, all annotated VRUs outside the blind spot zone are discarded. To compare the accuracy of our previous tracking-by-detection algorithm when using the delineated blind spot zone, we also need to discard all detections outside this blind spot zone.

Now suppose our framework detects a VRU at the border of the blind spot zone. If the annotation is located only just outside this blind spot zone (and thus is discarded), this detection would incorrectly count as a *false positive*. The same is true the other way around. If an annotation is located just inside the blind spot zone, and the detection just outside the zone (and again is discarded), this would account for an incorrect *false negative*. It is clear that, for validation purposes, a strict declaration of this blind spot zone – i.e. only keep annotations and detections which fall in this zone – is unwanted.

To overcome this problem we determined the accuracy in two steps. First, we compare *all detections in the zone* with *all annotations (including those outside*

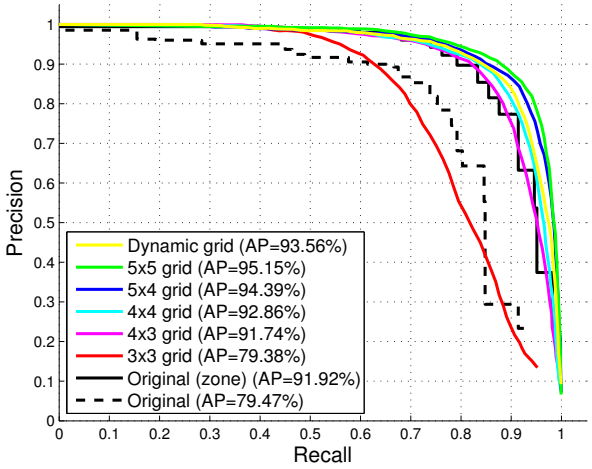


Figure 8.6: The accuracy of our algorithm when evaluated only in the delineated blind spot zone, for multiple grid sizes.

the zone). This yields all false positives, and part of the true positives. We ignore the false negatives since they now are meaningless. Next, we compare *all remaining annotations in the zone* with *all detections (including those outside the zone).* This yields all false negatives, and the second part of the true positives. We now ignore all false positives.

All the following accuracy experiments we now discuss were performed as mentioned above. Figure 8.6 displays these accuracy results for all grids, and our original tracking-by-detection implementation. The dashed black line indicates the accuracy of our VRU tracking-by-detection implementation which relies on initial search coordinates (as presented in chapter 7, section 7.2), without the use of our blind spot zone. The full black line displays the accuracy for this original implementation when only taking into account detections and annotations in the blind spot zone as discussed above. As seen, a significant gain in accuracy is achieved. When only detecting the VRUs in the delineated blind spot zone, our framework achieves an excellent average precision of 91.92%. This is due to multiple reasons. Annotations far outside the blind spot zone are difficult to detect, since they are very small. Furthermore, several different annotators were involved. Thus, the exact location behind the truck where the annotations start often significantly diverges between different sets. Due to the position of our initial search points in the tracking-by-detection framework, sometimes it was unfeasible to detect specific pedestrians or VRUs early enough.

As for the different grid sizes, interesting observations are noticed. The 3×3 grid evidently is not dense enough. Similar accuracy performance as to our original tracking-by-detection framework is achieved with a grid of 4×3 points. Apart from the deterministic calculation time, these grids have another significant advantage over our standard tracking-by-detection framework. There, a VRU that is being tracked might be lost if for multiple frames in a row no detection is found. No new search point is predicted in such cases. When using this default grid, tracks are much more easily recovered.

Note that both black curves (our original frameworks) are discretised (stepped). This is due to our tracking-by-detection approach. For these frameworks, we evaluated the accuracy performance for different discrete detection thresholds, since their temporal behaviour changes depending on each threshold. Indeed, future search points are predicted based on previous frames. For the different fixed search grids this is not the case. Here, we only perform detection at a single low threshold. For validation, we then vary this threshold in extremely small steps, resulting in more continuous PR curves.

Upon an increase in grid size, the accuracy further increases. For the 5×5 grid, we achieve our maximal average precision of 95.15%. However, evaluating 25 points has a negative impact on the calculation speed as discussed below. Our dynamic grid achieves an excellent accuracy (AP) of 93.56%. It significantly outperforms the linear 4×3 grid with an equal amount of grid points, thus indicating the effectiveness of the distortion specific positioning of these search points. Note that these accuracy results are still on single VRU detection capacity, the accuracy of the overarching blind spot alarm system is to be discussed in section 8.4.

Speed results when using grids

Figure 8.7 displays the processing speed of our framework when using the fixed detection grids in the delineated blind spot zone. The evaluation hardware remains identical as to previous chapters (Intel Xeon E5 CPU at 3.1 GHz). Our framework is mainly implemented in Matlab with time-consuming parts in both C and OpenCV. We present a sequential and a parallel implementation of our framework. Parallelisation was simply obtained by evaluating each grid point on a parallel CPU core. Since our test hardware consists of 32 cores, even at larger grid sizes (e.g. a 5×5 grid) each grid point is still evaluated on a separate core. When using for example a 4×3 grid (an identical size to the dynamic grid), we achieve a parallel processing speed of 6.2 frames per second. However, even when evaluated in parallel, increasing the size of the detection grid lowers the processing speed. Note that we still employ Matlab

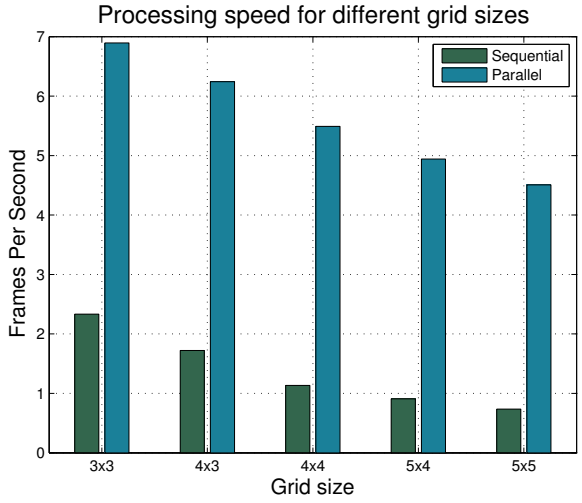


Figure 8.7: The processing speed of our framework when using a delineated blind spot zone and a fixed grid size.

which implies that multi-threaded processing and the data transfer between different threads is far from optimal. Indeed, this is seen when compared to our hybrid CPU/GPU implementation from chapter 4, section 4.3. Figure 4.24 on page 77 displays the grid size versus the number of frames per second for that optimal implementation. As seen, real-time processing speeds are achieved for the same grid sizes.

8.2.4 Conclusion

In this section we discussed a methodology to cope with the non-deterministic behaviour of our tracking framework. For this, we propose the use of specific detection grids within a delineated blind spot zone. We evaluated the accuracy and speed performance for several fixed grids, and a dynamic grid where the search points were positioned based on the degree of distortion.

Our experiments indicated that, with only a limited number of grid points, this approach achieves similar accuracy performance as our original tracking-by-detection framework. Moreover, our dynamic grid achieves even better accuracy performance at the computational complexity of a smaller fixed grid size. In the next section we present a second optimisation of our VRU tracking framework that extends the applicability from only adult pedestrians and bicyclists to children.

8.3 Detecting children

8.3.1 Introduction

Our perspective warping window approach performs pedestrian and bicyclist detection at a single scale only. This is a major advantage over traditional object detection methodologies which perform detection at all possible scales and locations in the feature pyramid. Indeed, restricting the number of search scales severely decreases the computation time. An example of such an approach is the branch and bound scheme presented by Lampert et al in [67]. However, only evaluating a specific scale has the disadvantage that objects that diverge from this specific scale are not detectable. To cope with this problem, we employed the deformable part models as object detection methodology. Our experiments indicated that, due to the deformation allowance of the parts of the detection model, this object detector is relatively invariant to slight variations in the estimated height of an object.

However, as children significantly deviate from our expected search scale, they currently are not detected. For this, we present an evident solution in subsection 8.3.2, and validate the gain in accuracy in subsection 8.3.3. We present our conclusions on this extension in subsection 8.3.4.

8.3.2 Approach

To cope with the detection of children, an evaluation of additional search scales is needed. Essentially two possible solutions towards this problem exist.

A first solution consists of the training of new detection models with smaller root sizes (thus, training a new model for a specific scale). The scale to which we warp the patches then remains fixed for both adults and children, and two different model sizes (or more if needed) are evaluated. Such an approach is analogous to the work of Benenson et al. [5], where the authors perform multiscale pedestrian detection by evaluating multiple models of different scales on a single image scale. As such, only a single-scale feature pyramid is constructed. However, training such multiple models is time-consuming. To tackle this, the authors reverse the insights of the FPDW detector [31]: only a limited number of models (one for each octave) are trained while the other models are approximated. Furthermore, obtaining training images for multiple models is not trivial. Large pedestrians are easily rescaled into smaller patches. However, blurring artifacts occur when small pedestrian image patches are upscaled.

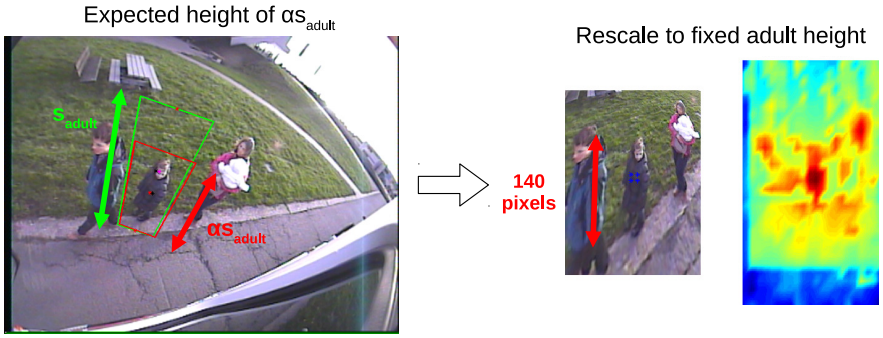


Figure 8.8: Our approach for the detection of children in our framework. We evaluate two warped versions of each image patch using two different scale factors.

A second solution simply consists of the evaluation of the same detection models on multiple (e.g. two) image patches. Each image scale is rewarped to the same optimal height of the detection models (140 pixels in our framework). However, a different scale factor, based on the position in the image, is used for both patches. This somewhat correlates to the traditionally used approach where one model is evaluated over multiple image scales. However, in our case we only need to evaluate a very limited number of scales since the deformable part models are invariant to slight height deviations.

When using the first approach in our framework, we only need to warp the image patches once with a single scale factor to the optimal height of 140 pixels, and evaluate multiple detection models for both adults and children. Although this is faster, additional detection models ought to be trained which requires significant training data and is time-consuming. Since in this section we focus on an improvement in accuracy, we opted to use the second approach. We rewarp each image patch twice to the optimal height of 140 pixels, with two different scale factors. For the first scale factor, we evidently employ the perspective transformation model for adult pedestrians. The second scale factor is simply calculated as the original adult scale factor divided by a fixed optimal value for children. To determine the optimal value, in our experiments we coin this parameter α. This parameter expresses the expected height of children as a percentage of the expected height of an adult. Our approach is visualised in figure 8.8, where we thus expect that children have a height of α times the expected height of an adult (indicated as s_{adult}) at that specific position. The right image shows the warped patch, which is rescaled to the optimal model height of 140 pixels, and the corresponding probability map generated as discussed in chapter 7, section 7.2.

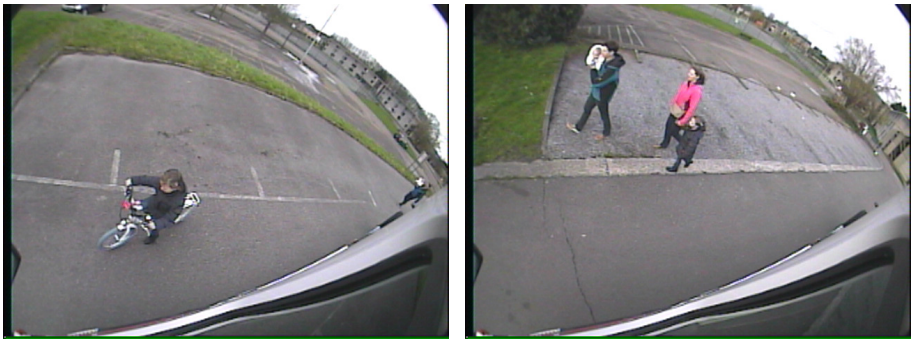


Figure 8.9: Two example frames of additionally recorded sequences containing children.

In the next section we present our experimental results for our proposed approach. We discuss how we determined the optimal value for this rescale factor α and analyse the effect on the detection accuracy of our framework.

8.3.3 Experiments and results

To validate our approach we recorded additional sequences including both adults and children. For this, four different sequences were recorded where a young family act out several scenarios. These sequences range from the parents walking in the blind spot zone e.g. hand in hand with their child, or where the child rides a bicycle. Figure 8.9 displays two example frames of these sequences. In total about 1800 frames were used for evaluation. These frames contain about 1200 adult annotations, and about 530 child annotations.

Our experiments were performed as follows. We evaluated the accuracy performance when including such an additional detection scale for all six detection grids proposed in section 8.2, and for multiple values of the scale factor α. To determine the optimal value, we varied this factor from 0.5 (= 50% of the calibrated adult height) to 0.95 (= 95% of the calibrated adult height) in a step size of 0.05. For each of both parameters (grid size and α) we determine the precision-recall curve for three implementations: our original single scale only implementation (i.e. $\alpha = 1.0$), the accuracy performance when only using the smaller α (child) scale, and a combination of both detection scales. This combination is performed as a simple *OR* operation, i.e. retaining all bounding boxes and only perform NMS. This is similar to how the vanilla deformable part model implementation combines the detections of the different components (i.e. viewpoints) in a mixture model [44, 46].

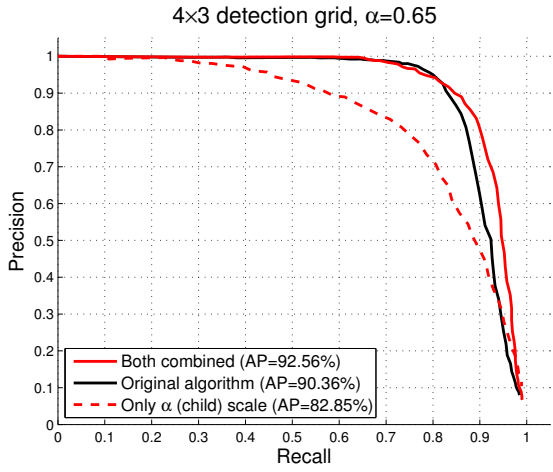
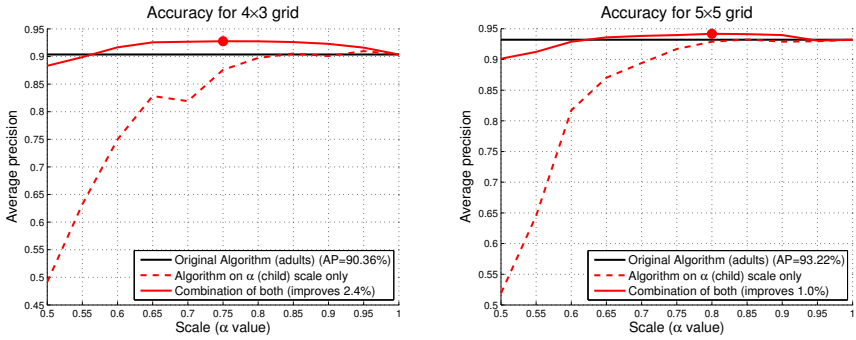


Figure 8.10: The accuracy results of our approach towards the detection of children, for a specific value of the grid size (4×3) and α value (0.65).

These experiments resulted in 60 different precision-recall curves (6 grids and 10 values for α), of which one of them (4×3 grid and $\alpha = 0.65$) is displayed in figure 8.10. The black curve indicates the accuracy of our standard – i.e. single (adult) scale – VRU detection framework on this new dataset. An average precision of 90.4% is achieved. The dashed red line indicates the accuracy of our framework when detection is performed on only the specific smaller scale (65% of the calibrated adult scale in this example). As seen, a combination of both effectively outperforms our initial single scale only implementation, reaching an AP of 92.6% (red curve). These experiments conclude that the inclusion of an additional scale increases the detection accuracy on these datasets containing children.

To get a concise overview of the influence on the accuracy performance when varying both parameters, we grouped the accuracy results from these 60 precision-recall curves as follows. For each grid we display the average precision (i.e. the area under the corresponding PR curve) for all three algorithms in function of the α value. Such resulting curves are visualised in figure 8.11, for two grid sizes: the 4×3 grid and the 5×5 grid. Here, the solid black line indicates the average precision of our original framework. The dashed red line indicates the average precision for the algorithm only on the specific scale, whereas the solid red line indicates the combined accuracy. For low α values, the combined accuracy is slightly lower (due to more false positives) while for high α values the detection scale diverges to the original detection scale. An



(a) 4×3 grid (b) 5×5 grid

Figure 8.11: The gain in accuracy when including an additional scale size (for two grid sizes).

optimal point is reached, indicated with the red dot. For the 4×3 grid, an increase of about 2.4% is achieved.

An interesting observation is noticed when analysing the curve for the 5×5 grid. If the size of the detection grid increases, the accuracy of our original algorithm increases, and the gain in accuracy when using the additional smaller search scale decreases. It seems that, if more detection points are allowed, our original framework is able to detect smaller pedestrians (children) without additional search points.

A similar, although reversed, observation was seen in figure 8.10. There, the α scale only algorithm already achieves good accuracy on this dataset which contains about two times more labelled adults as labelled children. Thus, this smaller scale framework manages to detect a considerable amount of adult pedestrians, although their scale significant differs.

This is explained as follows. Take for example the rescaled image patch in the right of figure 8.8. There, the adult pedestrian in the top right of this patch has an optimal detection height of about 140 pixels, and thus a high response in the probability map. In the original image frame (the left image in figure 8.8) we only searched for a small detection scale at a different location. However, due to the perspective transformation that we undo, this incidentally ensured that the adult pedestrian was transformed to an almost upright pedestrian with an optimal detection height. Including more grid points increases the probability that such events (and the other way around) occur. Including enough search points renders our framework thus inherently invariant to such scale variation and enables it to detect children without additional scales.

Note that only one distinctive child is used in these scenarios. As such, the experiments presented here serve as a proof of concept, indicating that our framework is able to detect these children. However, large scale experiments are needed to further validate this assumption. For example, for large datasets with more variation the evaluation of multiple subscales might be needed to achieve good detection accuracy.

Concerning computational complexity, including an additional search scale evidently increases the calculation time in a sequential implementation. However, such concerns can be tackled in multiple manners. This approach easily allows for a parallel implementation (i.e. where each search scale is evaluated in parallel). Furthermore, as discussed in subsection 8.3.2, here we only focused on an increase in accuracy. If needed, additional smaller models can be trained such that the features only need to be calculated once. This would significantly reduce the computation time. Indeed, our experiments indicated that the calculation of the features takes about 40% of the entire calculation time for each ROI, when evaluating a single detection model (see figure 7.13 on page 131).

8.3.4 Conclusion

In this section we proposed a method that – apart from adults – enables the detection of children. Initially we assumed that in our original framework this was not feasible, since we only perform detection at a single fixed scale. The deformable part models we employ as detection methodology are invariant to slight variations in height. However, children diverge too much from these expected search scales. For this, we experimented with the inclusion of an additional detection scale and evaluated the effect on the detection accuracy. Evidently, to validate this approach we extended our dataset with additional sequences containing children.

As our experiments indicate, the combination of multiple search scales increase the accuracy. However, as the grid size increases, the overall accuracy increases and the gain in accuracy induced by the additional scale decreases. Using a dense grid of search points already provides significant invariance to deviations in scale, and as such our framework even manages to detect children without the inclusion of this additional search scale.

In the next section we now elevate our original framework into a final alarm system, and present accuracy experiments as such.

8.4 System tests

8.4.1 Introduction

In previous chapters where we discussed the accuracy of our blind spot framework (and all sections above) our experimental results were performed on the level of *VRU tracking*. Indeed, our experiments validated the effectiveness of our framework with respect to the task of the detection and tracking of VRUs in the blind spot zone. For this, we achieved excellent accuracy results.

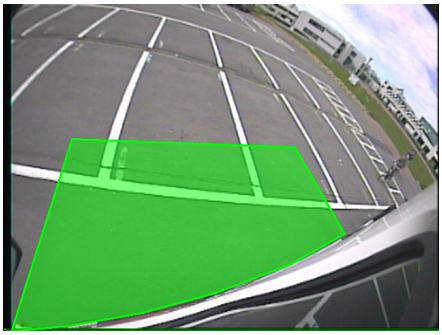
However, the final goal of this dissertation is the development of an active alarm system which automatically warns the truck driver when VRUs are present in the blind spot zone of the truck. Therefore, in this section we step away from the tracking-by-detection context, and propose experiments in which we effectively validate the performance of our entire framework when used as such an active alarm system. Our accuracy experiments now thus focus on *system level* tests.

Technical details of how these system tests were performed are discussed in subsection 8.4.2. We propose our experimental results in subsection 8.4.3. Finally, we present our conclusions in subsection 8.4.4.

8.4.2 Approach

We performed tests at system level as follows. We still maintain our complete detection system as discussed in chapter 7, section 7.2. However, we now step away from individual bounding boxes for each VRU, and generate an alarm if one or more VRUs are detected in the blind spot zone delineated as discussed in section 8.2. Figure 8.12 shows how our final alarm system currently displays the detection of VRUs in the blind spot zone. For each frame we determine if an alarm needs to be generated. For validation we thus classify each frame as a *true positive* (TP), *false positive* (FP), *false negative* (FN) and *true negative* (TN) as follows:

- *TP*: VRUs are present in the blind spot zone and an alarm is given.
- *FP*: no VRUs are present in the blind spot zone and an alarm is given.
- *FN*: VRUs are present in the blind spot zone and no alarm is given.
- *TN*: no VRUs are present in the blind spot zone and no alarm is given.



(a) No VRUs are detected in the blind spot zone.

(b) VRUs are detected in the blind spot zone.

Figure 8.12: Example frames of our final active alarm system. See `http://youtu.be/0S-uEPA_R5w` for a video.

Thus, for each frame we validate the effectiveness of our system. This is far from ideal, since no temporal information is taken into account. Furthermore, such evaluation metric is pessimistic when evaluating an alarm system such as in our application. Take for example the scenario where multiple true positive frames in a row occur when one or more VRU(s) enter the blind spot zone at the rear of the truck. In such cases the truck driver is warned. Now suppose – due to e.g. missed detections – a few consecutive frames are classified as *false negatives*. While not optimal, in real-world scenarios this is less of a concern since the truck driver was already warned. In such cases only a short interruption of the (audible) alarm signal would be noticed. However, the evaluation results presented here fail to take these considerations into account.

To further improve the accuracy of our system, we evaluated the integration of additional temporal information as follows. We aim to reject single (or short periods of) false positives. For this, we perform a sliding *majority voting* over the temporal frame per frame detection results. The exact size of this window (number of frames, N) is used as a parameter in our accuracy experiments, presented in the next section.

8.4.3 Experiments and results

The accuracy results as presented here are calculated over our final test set discussed in subsection 8.2.3. Thus, about 5500 frames were used for evaluation. Figure 8.13 displays the accuracy results of our final alarm system (solid black line), as compared to the accuracy of our original tracking-by-detection

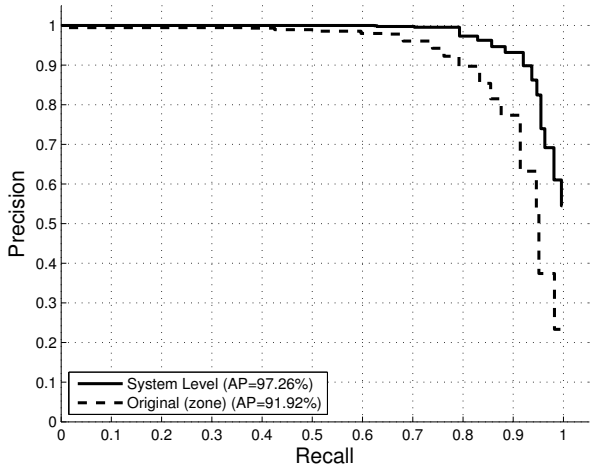


Figure 8.13: The accuracy of our final alarm system as compared to the original framework.

framework when only VRUs in the delineated blind spot zone are taken into account (discussed in section 8.2, indicated with the dashed black line). Recall that, for these system level tests we performed evaluation on a frame per frame basis.

Our active alarm system achieves an average precision of 97.26%. A significant accuracy improvement is realised over our standard tracking-by-detection framework where the accuracy is determined using all annotated VRUs in the delineated blind spot zone. This improvement is explained by the fact that the exact accuracy conditions are now shifted towards the system level. Take for example two pedestrians walking side by side in the blind spot zone, where one pedestrian is (partially) occluded by the other.

If in such case our standard tracking framework only detects the completely visible pedestrian, and fails to detect the occluded pedestrian, a false negative is counted resulting in a lower recall. Regarding our alarm system, finding only the completely visible pedestrian in the blind spot zone is sufficient, since this already generates an alarm (and thus the frame is regarded as being a *true positive*). A similar observation for false positives exist. If e.g. the occluded pedestrian was detected at the cost of a shifted bounding box (more than the overlap criterion of 50% used for evaluation), this would count as a false positive in the accuracy of our tracking framework. Again, in our final alarm system this false positive is not taken into account.

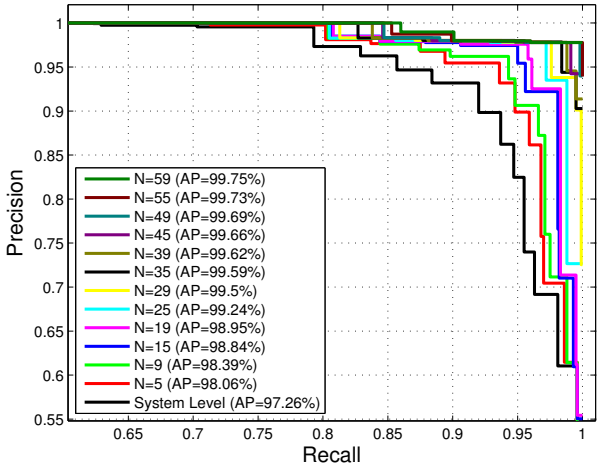


Figure 8.14: The accuracy of our final alarm system for different window sizes (N) of the majority voting.

As mentioned, to further increase the detection accuracy we employ a sliding majority voting over a window of N frames. For this, we simply take the most occurring classification in the window of the N previous frames as a final decision for that frame. Figure 8.14 displays the precision-recall curves of our alarm system for increasing sizes of this majority window (zoomed in on the top right corner). The black curve indicates our original algorithm ($N = 1$). An increase in size of this detection window evidently increases the accuracy.

For clarity, in figure 8.15 we plot the *majority window size* (N) versus the average precision of our final alarm system. Note that this curve in fact displays a trade-off between the detection accuracy of our system, and a measure of *latency*. If N increases, the average precision increases at the cost of a latency in the generation of the alarm signal.

The latency introduced by our majority voting scheme is determined as follows. Suppose our VRU detection framework is a perfect system. As such, each VRU that enters the blind spot zone is immediately detected in all consecutive frames. In such cases, the latency due to our majority voting equals $\frac{N+1}{2}$ frames. Evidently, in practice our VRU detection system is not perfect. This would avoid the need to use majority voting altogether. A worst-case latency of N frames is reached when our detection system only detects the VRU in alternating frames. A best-case latency of 1 frame is reached if a VRU is detected when he enters the frame, and due to false positives an alarm is given immediately.

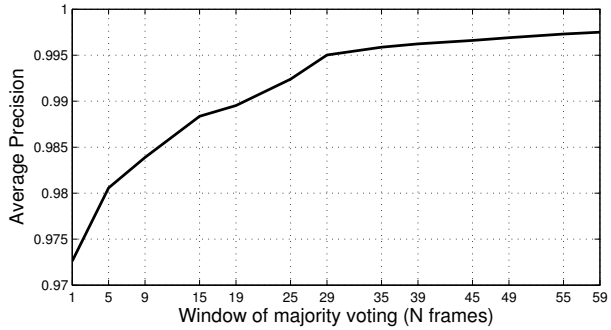


Figure 8.15: The average precision of our alarm system in function of the *majority window size* (N).

Take for example $N = 29$ frames. In such cases, our final alarm system achieves an average precision of about 99.5%. At a frame rate of 15 frames per second (and taken into account real-time detection performance), an alarm is generated after worst-case two seconds.

8.4.4 Conclusion

In this section we elevated our tracking-by-detection framework into a final alarm system. We proposed accuracy results when evaluating our system as such. Our system achieves an excellent average precision of 97.26% when evaluated on a frame per frame basis, raising to e.g. 99.5% when taking into account temporal smoothing. However, keep in mind that this evaluation methodology has several caveats as discussed above. To include temporal information we apply a sliding majority voting scheme. This allows for a trade-off between accuracy and latency of the alarm system.

In this section we validated the accuracy of our final active alarm system. In the next section we now discuss the usability of our alarm system in real-life scenarios.

8.5 Validating the usability of our final detection system

The acceptability and usability of an active alarm system as presented in the previous section depends on many considerations, thoroughly discussed in

chapter 2, section 2.5. These range from reliability, predictability, false error rate and so on. As mentioned there, an active blind spot detection system consists of at least two main components: the detection of the VRUs in the blind spot zone, and communicating this information to the truck driver. Although this latter component is of extreme importance towards the development of a commercial system, in this dissertation we focused on the first component: detecting the VRUs in an efficient manner. Regarding this technical detection component, here we now discuss if our proposed final alarm system meets all stringent design specifications of all three criteria which were given in chapter 2. We thus discuss the performance of our system with respect to the *throughput* (subsection 8.5.1), *latency* (subsection 8.5.2) and *accuracy* (subsection 8.5.3). With this analysis we will be able to address the question if our final active alarm system presented in this chapter is usable in real-life scenarios, as discussed in subsection 8.5.4. Finally our conclusions are given in subsection 8.5.5.

8.5.1 Throughput

The throughput that such a system should reach, defined in the number of frames per second, is easily quantifiable. Evidently, to be used in hard real-time scenarios our alarm system should be able to classify each detection frame at least as fast as the rate of which new frames need to be analysed. Typical commercial blind spot cameras achieve a frame rate of 15 frames per second.

The detection speed of our final active alarm system depends on the number of grid points. At e.g. a 3×3 detection grid, a processing speed of about 7 frames per second is achieved. Our final alarm system thus fails to meet this requirement. However, keep in mind that our final multiclass detection framework (used as a baseline in our final active alarm system) served as a proof of concept Matlab implementation. Thus, plenty of room for speed optimisation exists. Furthermore all algorithms presented in this dissertation were developed with low computational complexity in mind. Throughout multiple chapters in this dissertation we discussed several possible hardware implementation optimisations.

Indeed, for our initial warping window approach we developed such a highly optimised hybrid CPU/GPU implementation, and proved that real-time performance is achieved, even on embedded hardware (see chapter 4, section 4.3). As such, increasing the throughput of our final alarm system is only a matter of a more efficient implementation, e.g. no algorithmic redevelopment needs to be performed.

8.5.2 Latency

As discussed in chapter 2, the latency is defined as the time delay between the moment that VRUs enter the blind spot zone, and the moment that the system was able to raise an alarm. Evidently, the latency of such an alarm system should be minimal. As mentioned, research indicated that the worst-case reaction time of truck drivers equals about 1.5 seconds. As such, in the most extreme case, the truck driver should be warned before at least this reaction time added with the time needed to perform a stopping manoeuvre in advance. Quantifying the time this stopping manoeuvre takes is difficult, since it again depends on several factors such as weather conditions, the combined mass of the HGV and its truckload.

Due to the methodology of our active alarm system presented above, the maximally allowed latency depends on several factors. In our current final alarm system we defined that the zone in which VRU detection needs to be performed starts about 6.60 m behind the front of the truck's cabin. However, this is only a design parameter and as such can be increased – at the cost of higher computational complexity. Indeed, an increase of this detection zone allows for a larger latency. Furthermore, the allowed latency of our detection system is correlated with the relative speed difference between the VRUs and the truck. For pedestrians, this is of limited concern. However, for bicyclists this needs to be taken into account.

Additionally, when the *majority voting* scheme presented above is used, the latency is correlated with the accuracy. When increasing the number of frames (N) of the sliding majority window, the latency increases. The optimal value of this parameter depends on how the system is employed, as will be discussed in subsection 8.5.3.

Evidently, when no majority voting is used (i.e. $N = 1$) an immediate decision is taken for each individual frame. The latency thus equals the detection time for a single frame. If e.g. a detection speed of 15 frames per second is achieved, a latency of only 67 ms exist. However, this ideal scenario is based on offline processing of the image frames, and ignores the latency introduced by e.g. the frame grabber when capturing the image frames directly from the camera. For the remainder of this section we assume that this latter time is negligible, since in this dissertation we focused on a algorithmic proof of concept as opposed to a final commercial implementation. No experiments were performed with respect to the latency of different frame grabbers.

When majority voting is used to increase the accuracy, an increase in latency occurs. As an example, suppose $N = 5$. The (worst case) latency between a bicyclist who enters the frame at the rear of the truck, and the generation of

an alarm now takes 333 ms (at 15 frames per second and real-time detection performance). The relative speed difference between the bicyclist and the size of the detection zone needs to be taken into account to determine if this latency is small enough. Suppose a bicyclist with a velocity of 20 km/h approaches a truck taking a right-hand turn at 10 km/h. In such scenario, the time between entering the delineated detection zone and reaching the front right corner of the truck's cabin equals about 2.4 seconds. Thus, slightly more than 2 seconds remain for the truck driver to react in this particular situation.

As noticed, determining the minimal latency that such an alarm system requires is a difficult task. At best, when achieving real-time performance, our alarm system achieves a detection latency of 67 ms. In such cases, we achieve an average precision of 97.26% on our test set. If needed, the detection accuracy could further be increased at the cost of an increase in latency. If such an increase in latency is not allowed, the VRU detection zone could be extended.

8.5.3 Accuracy

Exact statistics concerning the number of allowed false alarms for active safety systems are difficult to obtain. In chapter 2 we discussed literature indicating that false alarm rates – also known as false positive rates (FPR) – of up to 2% are allowed. The false positive rate is defined as:

$$FPR = \frac{FP}{FP + TN} \tag{8.1}$$

As seen, the false positive rate includes the true negatives (TN), and as such highly depends on the exact dataset. In figure 8.16 we plot the recall of our final alarm system in function of N for three fixed values of the false positive rate (and thus precision): 0% (perfect precision), 1% (P=98.5%) and 1.8% (P=97.6%). This last FPR rate approximately equals the allowed rate in the literature. Therefore, for this FPR we list the exact figures in table 8.1. As seen, if we allow for a false positive alarm rate of 1.8%, our final alarm system achieves for e.g. $N = 5$ a recall of 89.3% (at a precision of 97.6%). Thus, when allowing for a false positive rate of 2%, about 90% of the time VRUs are in the blind spot zone, an alarm is generated. For increasing N this raises quickly to about 98% at 1.8% missed detections ($N = 29$). A perfect recall is achieved for $N = 55$.

However, defining the usability of a detection system solely on a specific value of the false positive rate is not optimal. For example, research indicated that a correlation exists between the acceptance of false alarms and the predictability of an alarm system. The false alarms are easier accepted if the circumstances

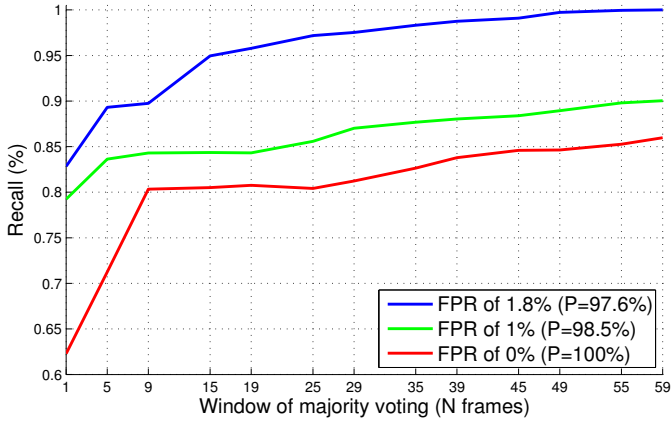


Figure 8.16: The recall of our alarm system in function of the *majority window size* (N) for three fixed values of the false positive rate.

Table 8.1: The recall of our final active alarm system for a false alarm rate of 1.8%.

N	**1**	**5**	**9**	**15**	**19**	**25**	**29**
Recall (%)	82.8	89.3	89.8	95.0	95.8	97.2	97.5

in which they occur are predictable. A safety system of which is known that false alarms occur during e.g. rainy conditions is easier accepted as compared to a safety system which produces a similar amount of false alarms at random moments.

Furthermore, a fixed value for the false positive rate implies a specific operating point (i.e. detection threshold) of our system. As a reminder, see figure 8.14 for the precision-recall curves of our active alarm system. At low detection thresholds most VRUs are detected, at the cost of many false alarms (i.e. high recall at low precision). At high detection thresholds only instances which have a high probability of being a VRU are detected (i.e. low recall at high precision. Defining the exact operating point of our active alarm system (and thus the trade-off between precision and recall) is difficult. This highly depends on *how* the system is used. If utilised as autonomous safety system (e.g. as an autonomous emergency braking system), a perfect accuracy is needed. Currently, our active alarm system fails to achieve perfect performance. However, if used as a decision support safety system for the driver, it offers a significant advantage if a high recall is achieved at (almost) perfect precision. For such scenarios the accuracy and usability of our alarm system is excellent.

Figure 8.17: An example of a typical false positive detection, leading to an increase in the false alarm rate.

Take for example the precision-recall curve in figure 8.14, for $N = 9$. If set for a perfect precision (i.e. no false alarms), a recall of 80.3% is achieved (as can be validated in figure 8.16 – red curve). Thus, in 80% of the time VRUs are present in the blind spot zone, our system generates an alarm. If a low false positive rate is allowed, the recall further increases. For example, at a recall of 85.4%, a false positive rate of only 1% is achieved. At these settings, about 34 frames of our entire dataset were classified as false positives, on a total of about 5500 frames (i.e. 0.62%). When measured in time units, about 27 seconds for each driven hour a false alarm is generated.

Keep in mind that these false positive frames are not distributed randomly over the entire dataset, since such single frame false positives are easily filtered out. The remaining false positives occur where an object in our dataset is misclassified as being a VRU for multiple frames in a row. After further examination, for these specific settings the main cause of the false positive frames was due to the false detection visualised in figure 8.17. There, due to shadows – with some imagination – an upper body-like appearance is seen. As will be discussed below, such false positives are easily eliminated when validated with e.g. an additional ultrasonic distance sensor. Furthermore, currently we assume that our alarm system continuously performs detection. However, commercial alarm systems are only activated at low speeds (using GPS) or when the driver signals a right-hand turn.

8.5.4 Discussion

As mentioned above, the applicability of such an active alarm system highly depends on *how* the system is employed. Indeed, for real-life scenarios a fixed detection threshold (and thus operating point on the accuracy curves) must be determined. However, determining this exact operating point of our active alarm system – and thus the trade-off between the false alarm rate and the detection accuracy – is difficult and remains open for discussion.

Although our final active alarm system as discussed above achieves excellent accuracy, it is not perfect yet. As such, currently it is not employable as autonomous safety system where such perfect accuracy is required. However, the accuracy results that our final system obtains renders it perfectly usable as decision support safety system. In such cases, the system is of great help if at (almost) perfect precision high recall values are achieved.

Indeed, when set for a perfect precision at low latency, our system achieves a recall of more than 80%. When low false positive rates are allowed (e.g. 1.8% – an acceptable value according to the literature), for the same low latency, the recall of our system increases to 90%. If an increase in latency is allowed, this recall value increases rapidly, as seen in figure 8.16. For example, for $N = 55$ a recall of 99.95% is achieved. At even higher latency values a perfect recall is obtained.

If such an increase in latency is unwanted, the blind spot zone should be extended such that it starts further behind the truck. This results in additional computational complexity. However, as we illustrated, our framework is able to achieve real-time processing speeds if an optimised implementation is developed.

Although our current alarm system already achieves excellent accuracy results, we indicate that further tests are needed to validate the effectiveness of our final active alarm system. In this dissertation we performed accuracy experiments on our own recorded dataset where we simulated several dangerous blind spot scenarios. The size of this dataset increased throughout multiple chapters. We aimed to keep the experimental setting as realistic as possible (using e.g. a real truck with genuine blind spot camera). The achieved accuracy results are promising when keeping a commercial system in mind, however large-scale in-the-field tests remain necessary to draw conclusions with respect to the usability of our final detection system. For example, difficult outdoor scenarios (e.g. fog or rain) and long-term test sequences are currently not included in our test set.

Additionally, several further optimisations are possible. For example, for the detection of VRUs in difficult light conditions an additional long-wave infrared

(LWIR) camera could be integrated in our system. Such a sensor would enable the detection of VRUs even in dark circumstances. However, at high outside temperatures or rainy conditions the temperature difference between VRUs and their surroundings is minimal. An integration of both LWIR and RGB data might be the best solution.

Currently our detection system solely relies on the blind spot camera as input sensor. To reduce the false positive rate, several commercial blind spot systems are only activated at low speeds (e.g. below 30 km/h, using GPS) or/and when the truck driver signals a right-hand turn. Such strategies are easily integratable in our detection scheme, and would allow for further optimisations concerning the latency versus accuracy trade-off. Furthermore, additional sensors could be integrated to validate a vision-based detected VRU. For example, ultrasonic distance sensors could be used to verify if an object is indeed present at the estimated location of the VRU. Such methodology would easily eliminate the false positive detection seen in figure 8.17. However, in this dissertation including these additional sensors is seen as future work.

8.5.5 Conclusion

In this section we evaluated if our final active alarm system achieved the requirements discussed in chapter 2. For this, we discussed three important design specifications: the throughput, latency and accuracy of our final alarm system.

Here we only focused on a proof of concept implementation obtaining high accuracy. Further work is needed to optimise the detection throughput of our final alarm system. Obtaining a higher throughput only requires a more efficient implementation, no algorithmic redevelopment is needed. Defining a final maximal value for the latency is difficult, since this depends on several external factors (such as the relative speed difference between the VRUs and the truck). However, our discussion indicated that our system is able to achieve adequate latency values.

A similar observation is found with respect to the required accuracy. For example, the exact allowed false error rate depends on the predictability of the system. Furthermore, the required accuracy strongly depends on how the system is used. We can conclude that, if used as a supportive safety system our final alarm system achieves excellent accuracy results and is usable as such.

8.6 Conclusion

In this chapter we elevated our final tracking-by-detection framework presented in chapter 7 into a final blind spot alarm system. For this, we first proposed two final extensions of our original framework.

First, we developed a method to eliminate the non-deterministic behaviour of our framework. For this, we defined different fixed-size grids, and evaluated their performance with respect to both the detection accuracy and processing speed. The use of only a small detection grid achieved similar accuracy results as our original algorithm, at fixed computational speeds. Additionally, we proposed a dynamic grid determined based on the degree of distortion at each position in the image. This dynamic grid further increases the detection accuracy at the same computational cost.

Next, we developed an approach that enables the detection of children. For this, we evaluated the inclusion of an additional detection scale. To evaluate our approach we evidently recorded additional datasets. Our experiments indicated that our approach increased the detection accuracy. We noticed however that, if a dense detection grid is used, due to the perspective transformation our system inherently is able to detect children without the inclusion of an additional scale.

Furthermore we presented a *system level* analysis of our final tracking-by-detection framework. For this, we converted our framework into a final active alarm system. When used as such, a frame per frame average precision of 97.26% was achieved. To further increase the accuracy we proposed the use of a *moving majority filtering*, achieving more than 99% average precision at still bearable latency. Indeed, this approach allows for a trade-off between accuracy and alarm latency. Finally, we discussed the usability of our active alarm system towards the use in real-life scenarios – and thus as a commercial system. For this, we extensively compared and discussed the final specifications that our system achieves with the specifications that such an active system requires to be usable in practice (which were given in chapter 2).

We showed that our system is able to meet these stringent requirements, when used as a decision support active alarm system. At perfect precision, a recall rate of more than 80% is achieved. At low false positive rates and slightly higher latency, our alarm system reaches recall values of up to three nines five. These results are promising when keeping a commercial system in mind. However, we note that further large-scale tests are needed to draw final conclusions with respect to the usability of our active alarm system. Furthermore, the inclusion of additional sensors (such as LWIR and ultrasonic distance sensors) allow to further boost the accuracy.

Chapter 9

Conclusion and Future work

9.1 Conclusion

The main goal of this dissertation was the development of an active alarm system for the blind spot zone of trucks, based solely on the blind spot camera images. Such a system should be able to detect vulnerable road users (VRUs) in the blind spot zone, and warn the truck driver of their presence in time to prevent fatal accidents. Several commercial systems exist (e.g. ultrasonic sensors or standard camera systems). However, research indicates that none of these seem to cope with the blind spot problem completely. For example, these ultrasonic sensors cannot distinguish e.g. pedestrians from other static objects such as traffic signs.

We believe that an alarm system which actively detects the VRUs in the blind spot camera images offers a possible solution to this problem. For this, we rely on *object* detection methodologies (in such context VRUs are regarded as being objects). However, this is a very challenging task. Indeed, such application inherently requires extremely stringent demands with respect to the detection accuracy, throughput and latency.

As the title of this dissertation suggests, this blind spot application is a specialisation of the more general detection of moving objects from a moving camera, and evidently relies on *hard* real-time behaviour. However, assuring hard real-time detection behaviour contradicts with the requirement for high accuracy. Specifically, object detection methodologies often require significant computational power to achieve high accuracy. As such, traditionally a trade-off exists between accuracy and throughput when only limited hardware is available.

In this dissertation we were able to develop a methodology (coined our *warping window approach*) that eliminates this trade-off. The advantage of this contribution is two-fold.

First, it allows for the detection of VRUs in challenging images where existing object detectors fail (due to the specific viewpoint and lens distortion). Second, this approach enables the use of highly accurate object detection methodologies which would otherwise be too time consuming. As such, we achieve excellent accuracy at real-time processing speeds. To validate this, we acquired a valuable dataset recorded with a genuine blind spot camera mounted on a real truck in which several dangerous blind spot scenarios were simulated. This dataset increased in size and complexity throughout this dissertation.

This initial methodology enabled the efficient detection and tracking of pedestrians in our blind spot camera images. Additionally, we proved that this methodology easily generalises to other scenarios with similar viewpoint.

Throughout this dissertation we presented contributions towards an increase in detection accuracy. We presented a methodology that enables the efficient combination of multiple pedestrian detectors, extended our initial approach to better model the specific distortion and developed a method to enable multiclass detection.

In the last chapter of this dissertation we integrated and further optimised the aforementioned methodologies and presented our final vision-based only active safety system for the blind spot zone.

We conclude that our final active alarm system manages to meet the stringent accuracy and latency demands required for such a system to be usable in practice. When used as a decision supportive safety system, depending on the latency, we report excellent recall rates of more than 80% at zero false positive rate. If low false positive rates are allowed – values of up to 2% are found in the literature – the recall rate further increases to 90%. If slightly higher latency is allowed, this quickly increases to more than three nines five.

The increase in latency can be accounted for by an increase in the size of the detection zone. This increases the computational complexity. However, we illustrated that our alarm system lends itself for real-time implementation. For this, no algorithmic redevelopment is needed. This is only a matter of a parallel, more optimised implementation.

9.2 Future work

Evidently, in each of the methodologies discussed in this dissertation further optimisations are possible. We first discuss possible technical improvements with respect to specific methodologies presented in this dissertation. We then present future work on our final active alarm system.

Several further optimisations of our *warping window approach* are possible. Currently, the system employs an offline calibration to determine the local distortion and transformation at each position in the image. For this, we extract data from labelled pedestrians (or a calibration board) homogeneously spread over the entire image region. This is tedious and time consuming, especially when aiming to develop commercial systems based on this approach.

For this, an automated calibration method should be implemented. These might range from simple methods, e.g. a pedestrian walking around in the image which is easily segmented out based on colour markers, to more advanced methods using e.g. a pedestrian detector in an offline exhaustive search over all scales and rotations, and only select the highest scoring detections as calibration instances.

Concerning the application of our warping window approach towards surveillance, to improve the detection accuracy on very difficult scenarios (e.g. long-term occlusions, people in chairs or people lying on the floor) several additional optimisations are possible. To cope with challenging poses an upper body detector or an evaluation at multiple rotations could be employed. For this, the rotation should be included in the tracker.

Currently we apply the deformable part model as object detector in our frameworks. As mentioned, this is due to its invariance with respect to slight variations in height. These are needed since we only perform detection at a single scale. Research indicated that rigid models (e.g. ACF) outperform these deformable part models with respect to the accuracy.

However, our experiments indicated that on only a single search scale, the accuracy of these rigid detection models is poor and as such we opted to use the deformable part model in our implementations. If future embedded hardware allows for the evaluation of multiple nearby search scales while remaining real-time, the integration of these rigid models becomes feasible and an increased accuracy might be obtained.

Furthermore, we only use appearance-based information in our framework for detection (apart from the motion information in the tracking-by-detection framework). For example, a dense optical flow field may be used to estimate the

ego-motion of the truck, and as such independent motion segmentation could be calculated. When combining this information with the pedestrian detection methodologies, the detection accuracy might further increase (and thus the false positive rate is reduced).

In our final probabilistic multiclass detection framework we currently combine the different detection models in a naive manner. A more profound combination of these models – using e.g. the combination methodology of chapter 6 – should be investigated. Furthermore, recall that additional models might be trained for children to reduce the computation time.

This combination methodology was based on two measures: the confidence and the complementarity. In future work a dynamic confidence measure (as opposed to a fixed value per detector used in this dissertation) based on image dependent features (e.g. contrast, height, texture and so on) might be used. This parameterised measure can then be used in two ways: either to further improve the detection accuracy, or in a select-out strategy where, during detection, the most appropriate pedestrian detector is selected for a specific image window.

The tracking-by-detection framework used in this dissertation could be improved in two ways. Currently, person re-identification is not performed, and we only select matching detections based on distance measures. Additional features (e.g. colour information) could be used to enable person re-identification. Furthermore, a multi-hypothesis tracker might be implemented in our final multiclass framework, where the weights of the particles are sampled in the calculated probability map.

Our final active alarm system achieves excellent accuracy specifications. These results are promising when keeping a commercial system in mind. However, although excellent performance is achieved, the step towards a true commercial system remains difficult. For this, a highly reliable and efficient real-time implementation needs to be developed. When developing such implementations for large-scale commercial use, many additional issues need to be addressed (such as stress-testing, cost efficiency and reliability of such a system).

In this dissertation we only performed research on the technical detection component of such an active alarm system. More research needs to be performed on the interaction between the truck driver and such a system. Keep in mind that in this dissertation we only presented a proof of concept alarm system. Currently, we performed validation on only a small dataset (when compared to real-life scenarios). We emphasise that large-scale tests remain needed to further validate the usability of our system in practice.

Indeed, our test set currently excludes difficult light or weather conditions. The integration of an additional long-wave infrared camera (LWIR) could be

a partial solution for this. The images returned from these cameras allow for pedestrian detection even in dark circumstances. However, only relying on this information source is not possible. For example, at high outside temperatures or rainy conditions the temperature difference between pedestrians and their surroundings is minimal. As such, the integration of these LWIR cameras with standard RGB cameras might be an optimal combination.

Our active alarm system was developed with only a vision-based sensor as input. To further increase the accuracy of our system, additional sensors could be integrated – as is done in several such commercial alarm systems. Indeed, existing systems are only activated below certain speeds (using GPS), and/or when the driver signals a right-hand turn. Furthermore, the presence of a detected VRU might be validated using e.g. ultrasonic distance sensors to reduce the number of false positives.

Bibliography

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face recognition with local binary patterns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 469–481. Springer, 2004.

[2] L. Akkermans. Technische hulpmiddelen ter voorkoming van dodehoekongevallen bij vrachtwagens (BIVV), 2009.

[3] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *In IEEE Transactions on Signal Processing (SP)*, 50(2):174–188, Feb. 2002.

[4] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Fast stixels computation for fast pedestrian detection. In *Proceedings of the European Conference on Computer Vision (ECCV), CVVT workshop*, pages 11–20, 2012.

[5] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool. Pedestrian detection at 100 frames per second. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2903–2910, 2012.

[6] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool. Seeking the strongest rigid detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3666–3673, Portland, Oregon, 2013.

[7] R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned? In *Proceedings of the European Conference on Computer Vision (ECCV), CVRSUAD workshop*, September 2014.

[8] R. Benenson, R. Timofte, and L. Van Gool. Stixels estimation without depth map computation. In *Proceedings of the IEEE International*

Conference on Computer Vision Workshops (ICCV Workshops), 2011, pages 2010–2017, 2011.

[9] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger. Review and evaluation of commonly-implemented background subtraction algorithms. In *Proceedings of the 19th IEEE International Conference on Pattern Recognition (ICPR)*, pages 1–4, 2008.

[10] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3457–3464, June 2011.

[11] BIVV. Via secura: ongevallen met vrachtwagens onder de loep, 2008.

[12] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 33(9):1820–1833, 2011.

[13] C. Bron and J. Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *In Communications of the Association for Computing Machinery (ACM)*, 16(9):575–577, 1973.

[14] CAVIAR project. The CAVIAR project: Context aware vision using image-based active recognition. `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`, 2005.

[15] C.-Y. Chan, F. Bu, and S. Shladover. *Experimental vehicle platform for pedestrian detection.* California PATH Program, Institute of Transportation Studies, University of California at Berkeley, 2006.

[16] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2014.

[17] H. Cho, P. E. Rybski, A. Bar-Hillel, and W. Zhang. Real-time pedestrian detection with deformable part models. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 1035–1042, August 2012.

[18] H. Cho, P. E. Rybski, and W. Zhang. Vision-based bicycle detection and tracking using a deformable part model and an EKF algorithm. In *Proceedings of the 13th IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1875–1880, 2010.

[19] G. Coley, A. Wesley, N. Reed, and I. Parry. Driver reaction times to familiar but unexpected events, 2008.

[20] Connekt. Dode hoek detectie- en signaleringssystemen (DDSS): Onderzoek naar de werking en mogelijkheden, 2010.

[21] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 886–893, 2005.

[22] S. De Beugher, G. Brône, and T. Goedemé. Automatic analysis of in-the-wild mobile eye-tracking experiments using object, face and person detection. In *Proceedings of the 9th International Conference on Computer Vision Theory And Applications (VISAPP)*, 2014.

[23] S. De Beugher, G. Brône, and T. Goedemé. Semi-automatic hand annotation making human-human interaction analysis fast and accurate. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, 2016.

[24] F. De Smedt. Pedestrian detection for real-life applications. In *PhD dissertation, KU Leuven*. 2015.

[25] F. De Smedt, D. Hulens, and T. Goedemé. On-board real-time tracking of pedestrians on a UAV. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 1–8, 2015.

[26] F. De Smedt, L. Struyf, S. Beckers, J. Vennekens, G. De Samblanx, and T. Goedemé. Is the game worth the candle? Evaluation of OpenCL for object detection algorithm optimization. In *Proceedings of the International Joint Conference on Pervasive and Embedded Computing and Communication Systems (PECCS)*, 2012.

[27] F. De Smedt*, K. Van Beeck*, T. Tuytelaars, and T. Goedemé. Pedestrian detection at warp speed: Exceeding 500 detections per second (* indicates equal contribution). In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 622–628, Portland, Oregon, 2013.

[28] F. De Smedt*, K. Van Beeck*, T. Tuytelaars, and T. Goedemé. The combinator: optimal combination of multiple pedestrian detectors (* indicates equal contribution). In *Proceedings of the 22nd International Conference on Pattern Recognition (ICPR)*, pages 3522–3527, Stockholm, Sweden, 2014.

[29] G. Debard, G. Baldewijns, T. Goedemé, T. Tuytelaars, and B. Vanrumste. Camera-based fall detection using a particle filter. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015.

[30] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 36(8):1532–1545, 2014.

[31] P. Dollár, S. Belongie, and P. Perona. The fastest pedestrian detector in the west. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 68.1–68.11, 2010.

[32] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 91.1–91.11, 2009.

[33] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features – addendum. 2009.

[34] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[35] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(4):743–761, 2012.

[36] W. Dubbink and van Luijk Henk. Bedrijfsgevallen: morele beslissingen van ondernemingen, 2006.

[37] C. Dubout and F. Fleuret. Exact acceleration of linear object detectors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 301–311, 2012.

[38] M. Eichner, M. Marin-Jimenez, A. Zisserman, and V. Ferrari. 2D articulated human pose estimation and retrieval in (almost) unconstrained still images. Technical report, 2010.

[39] M. Enzweiler and D. M. Gavrila. Monocular pedestrian detection: Survey and experiments. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(12):2179–2195, Dec. 2009.

[40] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[41] EU. Commission of the European Communities, European Road Safety Action Programme: mid-term review, 22 february 2006.

[42] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *In International Journal of Computer Vision (IJCV)*, 88(2):303–338, June 2010.

[43] P. Felzenszwalb, R. Girschick, and D. McAllester. Cascade object detection with deformable part models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2241–2248, 2010.

[44] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[45] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. `http://cs.brown.edu/~pff/latent-release4/`.

[46] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9):1627–1645, 2010.

[47] D. M. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *In International Journal of Computer Vision (IJCV)*, 73(1):41–59, 2007.

[48] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[49] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[50] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2012.

[51] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 437–446, 2015.

[52] R. B. Girshick, P. F. Felzenszwalb, and D. A. Mcallester. Object detection with grammar models. In *Proceedings of the Neural Information Processing Systems Conference (NIPS)*, pages 442–450, 2011.

[53] R. B. Girshick and J. Malik. Training deformable part models with decorrelated features. In *Proceedings of the IEEE International Conference on Computer Vision ICCV*, pages 3016–3023, 2013.

[54] E. Goubet, J. Katz, and F. Porikli. Pedestrian tracking using thermal infrared imaging. In *Proceedings of the SPIE Defense and Security Symposium (DSS)*, 2006.

[55] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *In Computing Research Repository (CoRR)*, 2015.

[56] D. Hoedemaeker, M. Doumen, M. De Goede, J. Hogema, R. Brouwer, and A. Wennemers. Modelopzet voor dodehoek detectie en signalerings systemen (DDSS), 2010.

[57] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015.

[58] J. Hosang, R. Benenson, and B. Schiele. How good are detection proposals, really? In *Proceedings of the British Machine Vision Conference (BMVC)*, 2014.

[59] J. Hosang, M. Omran, R. Benenson, and B. Schiele. Taking a deeper look at pedestrians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4073–4082, 2015.

[60] C. Huang, H. Ai, Y. Li, and S. Lao. Vector boosting for rotation invariant multi-view face detection. In *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 446–453, 2005.

[61] D. Hulens, T. Rumes, and T. Goedemé. Autonomous lecture recording with a PTZ camera while complying with cinematographic rules. In *Proceedings of the Conference on Computer and Robot Vision (CRV)*, pages 371–377, 2014.

[62] H. Jung, Y. Ehara, J. K. Tan, H. Kim, and S. Ishikawa. Applying MSC-HOG feature to the detection of a human on a bicycle. In *Proceedings of the 12th International Conference on Control, Automation and Systems (ICCAS)*, pages 514–517, 2012.

[63] R. E. Kalman. A new approach to linear filtering and prediction problems. *In Journal of Basic Engineering (JBE)*, 82(1):35–45, 1960.

[64] I. Knight. A study of the implementation of dir. 2007/38/EC on the retrofitting of blind spot mirrors to HGVs, 2011.

[65] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *In Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.

[66] L. Ladický, P. Sturgess, K. Alahari, C. Russell, and P. H. S. Torr. What, where and how many? Combining object detectors and CRFs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 424–437, 2010.

[67] C. Lampert, M. Blaschko, and T. Hoffmann. Efficient subwindow search: A branch and bound framework for object localization. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 31, pages 2129–2142, 2009.

[68] A. Leykin and R. Hammoud. Robust multi-pedestrian tracking in thermal-visible surveillance videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, pages 136–136, 2006.

[69] A. Leykin and R. Hammoud. Pedestrian tracking by fusion of thermal-visible surveillance videos. *In Machine Vision and Applications (MVA)*, 21(4):587–595, 2010.

[70] R. Lienhart and J. Maydt. An extended set of Haar-Like features for rapid object detection. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 900–903, 2002.

[71] H. Martensen. Themarapport vrachtwagenongevallen 2000 - 2007, 2009.

[72] M. Mathias, R. Benenson, R. Timofte, and L. Van Gool. Handling occlusions with Franken-classifiers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1505–1512, 2013.

[73] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool. Traffic sign recognition—how far are we from the solution? In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2013.

[74] C. Messom and A. Barczak. Fast and efficient rotated Haar-like features using rotated integral images. In *Proceedings of the Australian Conference on Robotics and Automation (ACRA)*, pages 1–6, 2006.

[75] W. Niewöhner, F. Berg, and F. Nicklisch. Innerortsunfälle mit rechts abbiegenden lastkraftwagen und ungeschützten verkehrsteilnehmern, 2005.

[76] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter. *In Journal on Image and Vision Computing (IVC)*, 21(1):99–110, 2003.

[77] E. Ohn-Bar and M. Trivedi. Fast and robust object detection using visual subcategories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 179–184, 2014.

[78] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *In Pattern Recognition (PR)*, 29(1):51–59, 1996.

[79] S. Orts-Escolano, J. Garcia-Rodriguez, V. Morell, M. Cazorla, J. Azorin, and J. M. Garcia-Chamizo. Parallel computational intelligence-based multi-camera surveillance system. *In Journal of Sensor and Actuator Networks (JSAN)*, 3(2):95–112, 2014.

[80] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of the 6th International Conference on Computer vision (ICCV)*, pages 555–562, 1998.

[81] D. Park, D. Ramanan, and C. Fowlkes. Multiresolution models for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 241–254. Springer, 2010.

[82] D. H. Parks and S. S. Fels. Evaluation of background subtraction algorithms with post-processing. In *Proceedings of the IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 192–199, 2008.

[83] M. Paul, S. M. Haque, and S. Chakraborty. Human detection in surveillance videos and its applications – a review. *In EURASIP Journal on Advances in Signal Processing (JASP)*, 2013(1):1–16, 2013.

[84] M. Pedersoli, J. Gonzalez, X. Hu, and X. Roca. Toward real-time pedestrian detection based on a deformable template model. *In IEEE Transactions on Intelligent Transportation Systems (ITS)*, 2013.

[85] M. Pedersoli, A. Vedaldi, and J. Gonzalez. A coarse-to-fine approach for fast deformable object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1353–1360, 2011.

[86] V. Philomin, R. Duraiswami, and L. Davis. Pedestrian tracking from a moving vehicle. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 350–355, 2000.

[87] V. Prisacariu and I. Reid. fastHOG - a real-time GPU implementation of HOG. Technical report, Department of Engineering Science, Oxford University, 2009.

[88] S. Puttemans, S. Vanwolputte, and T. Goedemé. Safeguarding privacy by reliable automatic blurring of faces in mobile mapping images. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*, 2016.

[89] F. Riguelle. Studie aangaande de efficientie van de anti-dodehoeksystemen, 2011.

[90] T. L. Robinson and W. Chislett. Commercial vehicle safety priorities - ranking of future priorities in the UK, 2010.

[91] G. Rogez, J. J. Guerrero, and C. Orrite. View-invariant human feature extraction for video-surveillance applications. In *Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 324–329, 2007.

[92] G. Rogez, C. Orrite, J. J. Guerrero, and P. H. S. Torr. Exploiting projective geometry for view-invariant monocular human motion analysis in man-made environments. *In Journal of Computer Vision and Image Understanding (CVIU)*, 120:126–140, 2014.

[93] G. Rogez, J. Rihan, J. J. Guerrero, and C. Orrite. Monocular 3D gait tracking in surveillance scenes. *In IEEE Transactions on Cybernetics*, 44(6):894–909, 2014.

[94] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *In International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[95] C. Schoon, M. Doumen, and D. de Bruin. De toedracht van dodehoekongevallen en maatregelen voor de korte en lange termijn, 2008.

[96] F. Seitner and A. Hanbury. Fast pedestrian tracking based on spatial features and colour. In *Proceedings of the Computer Vision Winter Workshop (CVWW)*, pages 105–110, 2006.

[97] O. Sidla, Y. Lypetskyy, N. Brandle, and S. Seer. Pedestrian detection and tracking for counting applications in crowded situations. In *Proceedings of the IEEE International Conference on Video and Signal Based Surveillance (AVSS)*, 2006.

[98] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[99] V. K. Singh, B. Wu, and R. Nevatia. Pedestrian tracking by associating tracklets using detection residuals. In *Proceedings of the IEEE Workshop on Motion and video Computing (WMVC), 2008.*, pages 1–8, 2008.

[100] P. Sudowe and B. Leibe. Efficient use of geometric constraints for sliding-window object detection in video. In *Proceedings of the International Conference on Computer Vision Systems (ICVS)*, pages 11–20, 2011.

[101] SWOV. SWOV-factsheet dodehoekongevallen, 2015.

[102] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[103] K. Takahashi, Y. Kuriya, and T. Morie. Bicycle detection using pedaling movement by spatiotemporal gabor filtering. In *Proceedings of the IEEE Region 10 Conference (TENCON)*, pages 918–922, 2010.

[104] W. Tian and M. Lauer. Fast cyclist detection by cascaded detector and geometric constraint. In *Proceedings of the IEEE 18th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1286–1291, 2015.

[105] K. Van Beeck, F. De Smedt, T. Goedemé, and T. Tuytelaars. Towards robust automatic detection of vulnerable road users: Monocular pedestrian tracking from a moving vehicle. In *ATINER 7th Annual International Conference on Computer Science and Information Systems*, Athens, Greece, 2011.

[106] K. Van Beeck and T. Goedemé. Real-time pedestrian detection in a truck's blind spot camera. In *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 412–420, Angers, France, 2014.

[107] K. Van Beeck and T. Goedemé. Efficient multiclass object detection: Detecting pedestrians and bicyclists in a truck's blind spot camera. In *Proceedings of the 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Kuala Lumpur, Malaysia, 2015.

[108] K. Van Beeck and T. Goedemé. Fast and accurate pedestrian detection in a truck's blind spot camera. In *Pattern Recognition Applications and Methods*, volume 9443 of *Lecture Notes in Computer Science*, pages 179–195. Springer International Publishing, 2015.

[109] K. Van Beeck and T. Goedemé. Real-time accurate pedestrian detection and tracking in challenging surveillance videos. In *Proceedings of the 10th International Conference on Computer Vision Theory And Applications (VISAPP)*, pages 325–334, Berlin, Germany, 2015.

[110] K. Van Beeck and T. Goedemé. The automatic blind spot camera: a vision-based active alarm system. In *Proceedings of Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD, ECCV workshop)*, Amsterdam, The Netherlands, 2016.

[111] K. Van Beeck and T. Goedemé. Pedestrian detection and tracking in challenging surveillance videos. In *Computer Vision, Imaging and Computer Graphics Theory and Applications*, volume 598 of *Communications in Computer and Information Science*, chapter 19, pages 356–373. Springer International Publishing, 2016.

[112] K. Van Beeck, T. Goedemé, and T. Tuytelaars. Towards an automatic blind spot camera: Robust real-time pedestrian tracking from a moving camera. In *Proceedings of the 12th IAPR Conference on Machine Vision Applications (MVA)*, pages 528–531, Nara, Japan, 2011.

[113] K. Van Beeck, T. Goedemé, and T. Tuytelaars. A warping window approach to real-time vision-based pedestrian detection in a truck's blind spot zone. In *Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 561–568, Rome, Italy, 2012.

[114] K. Van Beeck, T. Goedemé, and T. Tuytelaars. Real-time vision-based pedestrian detection in a truck's blind spot zone using a warping window approach. In *Informatics in Control, Automation and Robotics*, volume 238 of *Lecture Notes in Electrical Engineering*, chapter 16, pages 251–264. Springer International Publishing, 2014.

[115] C. Van den Meersschaut. Dode hoek: samen kunnen we er iets aan doen! brochure voor vrachtwagenchauffeurs en ondernemening die vrachtwagens gebruiken.

[116] Van Dievel Transport. 75 jaar duurzaam transport, 2014.

[117] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 511–518, 2001.

[118] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 63, 2003.

[119] D. M. Vo, L. Jiang, and A. Zell. Real time person detection and tracking by mobile robots using RGB-D images. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 689–694, 2014.

[120] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical report, 1995.

[121] C. Wojek, S. Walk, and B. Schiele. Multi-cue onboard pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 794–801, 2009.

[122] P. Xu, F. Davoine, and T. Denoeux. Evidential combination of pedestrian detectors. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2014.

[123] K. Yamaguchi. OpenCV Mex toolbox. `http://github.com/kyamagu/mexopencv`.

[124] J. Yan, X. Zhang, Z. Lei, S. Liao, and S. Li. Robust multi-resolution pedestrian detection in traffic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3033–3040, 2013.

[125] S. Yuki, D. Daisuke, K. Yasutomo, I. Ichiro, and M. Hiroshi. Detector ensemble based on false positive mining for pedestrian detection. In *Proceedings of the 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Kuala Lumpur, Malaysia, 2015.

[126] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. How far are we from solving pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[127] S. Zhang, R. Benenson, and B. Schiele. Filtered channel features for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1751–1760, 2015.

[128] F. Zhao and J. Li. Pedestrian motion tracking and crowd abnormal behavior detection based on intelligent video surveillance. *In Journal of Networks (JNW)*, 9(10):2598–2605, 2014.

[129] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *In journal of Pattern Recognition Letters (PRL)*, 27(7):773–780, 2006.

List of publications

International conference publications

[1] K. Buys, K. Van Beeck, T. Goedemé, and H. Bruyninckx. Towards high speed 6DoF inertial-visual SLAM: robust image feature detection. In *Proceedings of the IARP Workshop on Robotics for Risky Interventions and Environmental Surveillance (RISE)*, Brussels, Belgium, 2009.

[2] K. Van Beeck, F. Heylen, J. Meel, and T. Goedemé. Comparative study of model-based hardware design tools. In *Proceedings of the 4th European Conference on the Use of Modern Information and Communication Technologies (ECUMICT)*, Ghent, Belgium, 2010.

[3] K. Van Beeck, T. Goedemé, and T. Tuytelaars. Towards an automatic blind spot camera: Robust real-time pedestrian tracking from a moving camera. In *Proceedings of the 12th IAPR Conference on Machine Vision Applications (MVA)*, pages 528–531, Nara, Japan, 2011.

[4] K. Van Beeck, F. De Smedt, T. Goedemé, and T. Tuytelaars. Towards robust automatic detection of vulnerable road users: Monocular pedestrian tracking from a moving vehicle. In *ATINER 7th Annual International Conference on Computer Science and Information Systems*, Athens, Greece, 2011.

[5] L. Van den Heuvel, K. Van Beeck, and T. Goedemé. A comparison of optical flow techniques for a blind spot camera. In *Proceedings of the IADIS International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP)*, pages 331–334, Rome, Italy, 2011.

[6] G. Brône, B. Oben, K. Van Beeck, and T. Goedemé. Towards a more effective method for analyzing mobile eye-tracking data: Integrating

gaze data with object recognition algorithms. In *Proceedings of the 1st International Workshop on Pervasive Eye Tracking and Mobile Eye-based interaction (PETMEI)*, pages 53–56, Beijing, China, 2011.

[7] K. Van Beeck, T. Goedemé, and T. Tuytelaars. A warping window approach to real-time vision-based pedestrian detection in a truck's blind spot zone. In *Proceedings of the 9th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pages 561–568, Rome, Italy, 2012.

[8] F. De Smedt*, K. Van Beeck*, T. Tuytelaars, and T. Goedemé. Pedestrian detection at warp speed: Exceeding 500 detections per second (* indicates equal contribution). In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 622–628, Portland, Oregon, 2013.

[9] K. Van Beeck and T. Goedemé. Real-time pedestrian detection in a truck's blind spot camera. In *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 412–420, Angers, France, 2014.

[10] F. De Smedt*, K. Van Beeck*, T. Tuytelaars, and T. Goedemé. The combinator: optimal combination of multiple pedestrian detectors (* indicates equal contribution). In *Proceedings of the 22nd International Conference on Pattern Recognition (ICPR)*, pages 3522–3527, Stockholm, Sweden, 2014.

[11] K. Van Beeck and T. Goedemé. Real-time accurate pedestrian detection and tracking in challenging surveillance videos. In *Proceedings of the 10th International Conference on Computer Vision Theory And Applications (VISAPP)*, pages 325–334, Berlin, Germany, 2015.

[12] K. Van Beeck and T. Goedemé. Efficient multiclass object detection: Detecting pedestrians and bicyclists in a truck's blind spot camera. In *Proceedings of the 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Kuala Lumpur, Malaysia, 2015.

[13] D. Hulens, K. Van Beeck, and T. Goedemé. Fast and accurate face orientation measurement low-resolution images on embedded hardware. In *Proceedings of the 11th International Conference on Computer Vision Theory And Applications (VISAPP)*, Rome, Italy, 2016.

[14] K. Van Beeck and T. Goedemé. The automatic blind spot camera: a vision-based active alarm system. In *Proceedings of Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD, ECCV workshop)*, Amsterdam, The Netherlands, 2016.

Book chapter publications

[1] K. Van Beeck, T. Goedemé, and T. Tuytelaars. Real-time vision-based pedestrian detection in a truck's blind spot zone using a warping window approach. In *Informatics in Control, Automation and Robotics*, volume 238 of *Lecture Notes in Electrical Engineering*, chapter 16, pages 251–264. Springer International Publishing, 2014.

[2] K. Van Beeck and T. Goedemé. Fast and accurate pedestrian detection in a truck's blind spot camera. In *Pattern Recognition Applications and Methods*, volume 9443 of *Lecture Notes in Computer Science*, pages 179–195. Springer International Publishing, 2015.

[3] K. Van Beeck and T. Goedemé. Pedestrian detection and tracking in challenging surveillance videos. In *Computer Vision, Imaging and Computer Graphics Theory and Applications*, volume 598 of *Communications in Computer and Information Science*, chapter 19, pages 356–373. Springer International Publishing, 2016.

Journal publications

[1] W. Meeus, K. Van Beeck, T. Goedemé, J. Meel, and D. Stroobandt. An overview of today's high-level synthesis tools. *Design Automation for Embedded Systems*, 16(3):31–51, 2012.

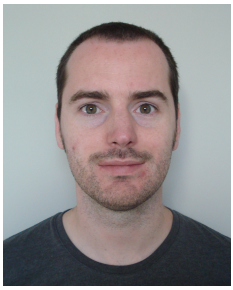
Abstract publications

[1] K. Van Beeck, T. Goedemé, and T. Tuytelaars. The active blind spot camera: hard real-time recognition of moving objects from a moving camera. In *European conference on the use of modern information and communication technologies (ECUMICT)*, Ghent, Belgium, 2012.

[2] K. Van Beeck, T. Goedemé, and T. Tuytelaars. The active blind spot camera: hard real-time recognition of moving objects from a moving camera. In *European conference on the use of modern information and communication technologies (ECUMICT)*, Ghent, Belgium, 2014.

[3] K. Van Beeck and T. Goedemé. The active blind spot camera: hard real-time recognition of moving objects from a moving camera. In *Research Day FIIW-ESAT/CW*, Leuven, Belgium, 2015.

Curriculum Vitae



Kristof Van Beeck was born on the 25th of November 1986 in Lier, Belgium. He obtained a master degree in Industrial Sciences - Electronics-ICT at the *Hogeschool Voor Wetenschap & Kunst - Campus De Nayer* (now a KU Leuven Technology Campus) in 2008. During his master thesis, titled *Implementation of the SURF algorithm in an Embedded Platform*, he gained his first experience concerning computer vision algorithms and their integration on embedded hardware, under the supervision of Prof. Toon Goedemé. Intrigued by the fascinating opportunities that embedded computer vision enables, he decided to start as a researcher on a two-year project (called *Fast-ProMoCo*) at campus De Nayer under the supervision of Prof. Jan Meel. In this company-driven research project he evaluated the effectiveness and usability of several model-based hardware design tools. Around that time, the EAVISE research group was founded which specifically focuses on applications of advanced computer vision and artificial intelligence. Within this context of applying state-of-the-art computer vision techniques to very specific applications, he started his doctoral research in 2010 under the supervision of Prof. Toon Goedemé. In this work towards a PhD degree he focused on the detection of moving objects from a moving camera, more specifically the reliable and hard real-time detection of vulnerable road users in blind spot camera images.

FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
EMBEDDED ARTIFICIALLY INTELLIGENT VISION ENGINEERING (EAVISE)
Jan De Nayerlaan 5
B-2860 Sint-Katelijne-Waver
kristof.vanbeeck@kuleuven.be
http://www.eavise.be